VALSTS REĢIONĀLĀS ATTĪSTĪBAS AĢENTŪRA

VALSTS INFORMĀCIJAS SISTĒMU SAVIETOTĀJA (VISS) UN VIENOTĀ VALSTS UN PAŠVALDĪBU PAKALPOJUMU PORTĀLA WWW.LATVIJA.LV PILNVEIDOŠANA UN UZTURĒŠANA

E-PAKALPOJUMA IZVEIDOŠANAS CEĻVEDIS LATVIJA.LV PORTĀLAM

PROGRAMMĒTĀJA ROKASGRĀMATA

VRAA-13_7_17_41-VISS_2016-LVP_WZD_3-PR

26.09.2022. versija 1.17

Rīgā 2022

Satura rādītājs

Attēlu saraksts

1. Ievads

Viena no Latvija.lv portāla (turpmāk LVP) funkcijām ir piedāvāt iedzīvotājiem e-pakalpojumus. E-pakalpojumus izstrādā kā patstāvīgus lietojumus, kurus integrē LVP vidē.

1.1. Dokumenta nolūks

Dokumenta nolūks ir iepazīstināt izstrādātājus ar LVP e-pakalpojumu veidošanas principiem, integrāciju ar LVP, piedāvātajām bibliotēkām, kā arī nodrošināt citu e-pakalpojumu izstrādes uzsākšanai nepieciešamo informāciju.

Dokumentā iekļauts arī izstrādes vides apraksts un tās uzstādīšanas instrukcija.

1.2. Darbības sfēra

LVP ir publiska vietne e-pakalpojumu izmitināšanai. Šis dokuments paredzēts e-pakalpojumu izstrādātājiem.

1.3. Termini un pieņemtie apzīmējumi

Apzīmējumu un terminu vārdnīca pieejama dokumentā [1].

Šī dokumenta ietvaros termins CDN tiek lietots, lai aprakstītu tīmekļa lietojumprogrammu, kuras vienīgais uzdevums ir nodrošināt resursus citām lietojumprogrammām (attēli, JS un CSS datnes).

1.4. Saistība ar citiem dokumentiem

Dokuments ir izstrādāts, balstoties uz šādiem dokumentiem:

[1] „Valsts informācijas sistēmu savietotāja, Latvijas valsts portāla www.latvija.lv un elektronisko pakalpojumu izstrāde un uzturēšana". Iepirkuma priekšmeta 3.daļa - VISS un portāla jaunu un esošo moduļu papildinājumu izstrāde, ieviešana, garantijas apkalpošana un uzturēšana saskaņā ar tehnisko specifikāciju. Terminu un saīsinājumu indekss. (VRAA-6_15_11_58-VISS_2010-TSI).

[2] „Valsts informācijas sistēmu savietotāja, Latvijas valsts portāla www.latvija.lv un elektronisko pakalpojumu izstrāde un uzturēšana". Iepirkuma priekšmeta 3.daļa - VISS un portāla jaunu un esošo moduļu papildinājumu izstrāde, ieviešana, garantijas apkalpošana un uzturēšana saskaņā ar tehnisko specifikāciju. Elektronisko dokumentu krātuve: ārējās saskarnes. Programmatūras projektējuma apraksts. (VRAA-6_15_11_58-VISS_2010-EDK_AS-PPA).

[3] Par Valsts informācijas sistēmu savietotāja, Latvijas valsts portāla www.latvija.lv un elektronisko pakalpojumu izstrāde un uzturēšana. 3.daļa „VISS un Portāla jaunu un esošo moduļu papildinājumu izstrāde, ieviešana, garantijas apkalpošana un uzturēšana saskaņā ar tehnisko specifikāciju". E-pakalpojumu arhitektūras izstrāde. Vadlīnijas. (VRAA-13_7_17_41-VISS_2016-EPAK_ARH_3-VDL).

[4] „Valsts informācijas sistēmu savietotāja, Latvijas valsts portāla www.latvija.lv un elektronisko pakalpojumu izstrāde un uzturēšana". Iepirkuma priekšmeta 3.daļa - VISS un portāla jaunu un esošo moduļu papildinājumu izstrāde, ieviešana, garantijas apkalpošana un uzturēšana saskaņā ar tehnisko specifikāciju. VISS Drošības talonu serviss. Programmētāja rokasgrāmata. (VRAA-6_15_11_58-VISS_2010-DTS-PR).

[5] "Valsts informācijas sistēmu savietotāja (viss) un vienotā valsts un pašvaldību pakalpojumu portāla www.latvija.lv pilnveidošana un uzturēšana". Elektronisko dokumentu krātuve. Programmatūras ārējo saskarņu projektējums. (VRAA-13_7_17_41-VISS_2016-EDK-PPA_AS).

[6] Valsts informācijas sistēmu savietotāja (VISS) un Vienotā valsts un pašvaldību pakalpojumu portāla www.latvija.lv pilnveidošana un uzturēšana. 3.daļa "VISS un Portāla jaunu un esošo moduļu papildinājumu izstrāde, ieviešana, garantijas apkalpošana un uzturēšana saskaņā ar tehnisko specifikāciju". Datu apmaiņas izveides vadlīnijas. Vadlīnijas. (VRAA-13_7_17_41-VISS_2016-D_APM-VDL).

[7] Valsts informācijas sistēmu savietotājs (VISS) un Vienotā valsts un pašvaldību pakalpojumu portāla www.latvija.lv pilnveidošana un uzturēšana. Maksājumu modulis un Iedzīvotāju maksājumu nodrošināšana Integrācijas instrukcija (VRAA-13_7_17_41 -VISS_2016-MM-II).

[8] Adrešu meklēšanas komponentes attīstība un uzturēšana. AMK 1.IP izstrāde Integrācijas apraksts (VRAA-13-7/20/143-AMK-IAPR).

[9] VISS sistēmas žurnāls. Koplietojuma bibliotēku apraksts (VRAA-13_7_17_41-VISS_2016-VISS_ZUR-KBA).

[10] VRAA kubernetes infrastruktūras komponenšu piegāžu procesa organizācija (VRAA-K8S_INFR-DeleveryProcess-VDL)

[11] Valsts informācijas sistēmu savietotāja (VISS) un Vienotā valsts un pašvaldību pakalpojumu portāla www.latvija.lv pilnveidošana un uzturēšana. Integrācijas instrukcija. (VRAA-13_7_17_41-VISS_2016-EDK-II).

1.5. Dokumenta pārskats

Dokumentu veido šādi nodalījumi:

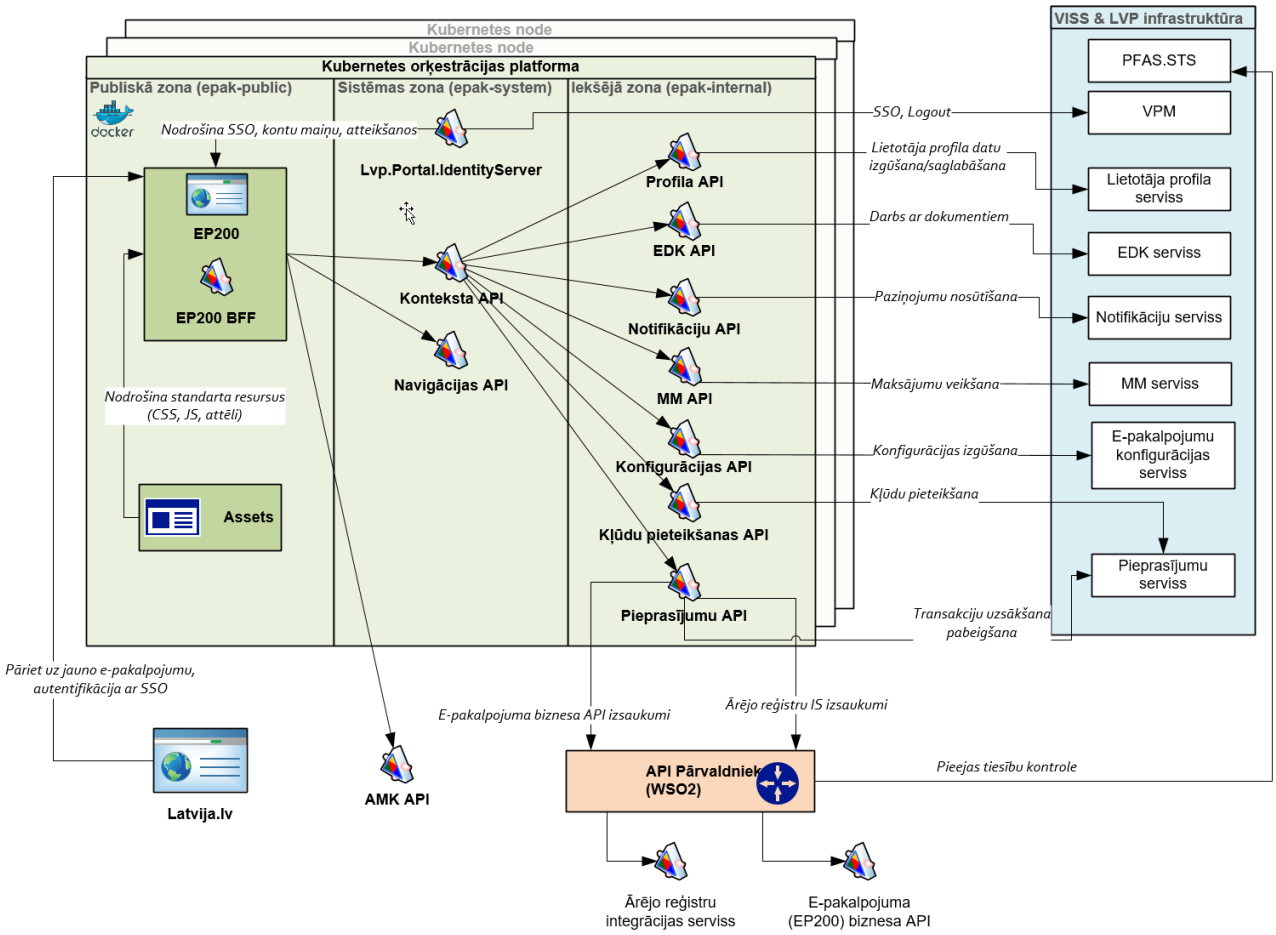
- „Ievads" – iekļauta informācija par dokumenta vispārējo struktūru, darbības sfēru, nolūku, dokumentā izmantotajiem terminiem un apzīmējumiem, kā arī par saistību ar citiem dokumentiem.
- „Risinājuma arhitektūra" – vispārīgs apraksts par e-pakalpojumu integrāciju ar LVP. Sākumpunkts izstrādātājiem, lai saprastu kopainu.
- "Izstrādes vides sagatavošana" – vides prasību apraksts un tās sagatavošana.
- „Jauna e-pakalpojuma izveidošana" – instrukcija jauna e-pakalpojuma izstrādes uzsākšanai, izmantojot piedāvāto sagatavi.

- “E-paklpojumu projekts” – React un MVC projekta struktūras apraksts.
- „Bibliotēkas e-pakalpojumu izstrādei” – e-paklpojumu izstrādātājiem pieejamās bibliotēkas un to izmantošanas instrukcijas.
- „Servisi e-pakalpojumu izstrādei” – izstrādātājiem pieejamie servisi.
- “E-pakalpojumu koneineru izstrāde un piegāde” – instrukcija e-pakalpojuma dokeru sagatavošanai un automatizētai piegādei.
- “E-pakalpojumu piemēri” – e-paklpojumu izstrādātājiem pieejamo e-pakalpojumu piemēru apraksts.
- “Biežāk sastopamās problēmas un to risinājumi” – problēmu apraksti un risinājumi.

2. Risinājuma arhitektūra

E-pakalpojumu platformas risinājumu var iedalīt četrās daļās:

1. E-pakalpojumu grafiskās saskarnes un to apkalpojošais slānis – var tikt veidoti kā viens vai atsevišķi konteineri, kuri tiek izmitināti publiskajā zonā (epak-public vārdtelpā).
2. E-pakalpojumu biznesa servisi – REST servisi, kas nodrošina konkrētajam e-pakalpojumam paredzēto biznesa loģiku. Tie tiek reģistrēti API pārvaldniekā.
3. E-pakalpojumu platformas API – REST servisi, kas nodrošina pieeju pie VISS infrastruktūras servisu nodrošinātās funkcionalitātes, detalizētu API aprakstu un izmantošanas instrukciju skatīt Servisi e-pakalpojumu izstrādei nodaļā.
4. LVP un VISS infrastruktūras servisi - tiešā veidā e-pakalpojumos nav paredzēts izmantot.



1.attēls. E-pakalpojumu platformas risinājuma arhitektūra

2.1. Konteksta API

Konteksta API darbojas kā proxy piekļūšanai pie iekšējās zonas API un nodrošina darbu ar lietotāja sesiju:

- Vērtības saglabāšanu/aktualizēšanu sesijā;
- Sesijā saglabātās vērtības nolasīšanu;
- Visu sesijā saglabāto vērtību nolasīšanu;
- Sesijā saglabātās vērtības dzēšanu.

Sesija dod iespēju veikt īslaicīgu datu glabāšanu (1h konfigurējams sistēmas līmenī), kas nepieciešami konkrētā e-pakalpojuma izpildei. Tajā nav paredzēts glabāt lielapjoma datus, jo tas var izraisīt konteksta API konteinera atmiņas limitu pārsniegšanu un izraisīt tā piespiedu apturēšanu.

Lai samazinātu pieprasījumu skaitu un pārsūtamo datu apjomu sesija tiek integrēta citos 'iekšējās zonas' API, piemēram Pieprasījumu API sesijā saglabā izveidoto transakciju, bet Notifikāciju API to nolasa. Sesijā var tikt saglabāta tikai viena transakcija.

Sesijas funkcionalitāte pieejama izmantojot e-pakalpojumu ietvara *LvpContext.SessionProperties* aizmugursistēmas metodes (skat. 7.1. nodaļu).

2.2. Pieprasījumu API

Pieprasījumu API nodrošina:

- E-pakalpojuma transakcijas izveidi;
- E-pakalpojuma transakcijas noslēgšanu;
- API pārvaldniekā reģistrēta, savas vai citas iestādes, integrācijas servisa izsaukšanu. Veicot jaunu e-pakalpojumu izstrādi izmantojot e-pakalpojumu platformu nav paredzēts veidot jaunus integrācijas servisus, tā vietā jāveido REST servisi [6], kuri jāreģistrē API pārvaldniekā atbilstoši datu apmaiņu vadlīnijām [6], Lai no e-pakalpojuma izsauktu API pārvaldniekā reģistrētu integrācijas servisu e-pakalpojums nav jāreģistrē API pārvaldniekā (Store) kā klienta lietojums;
- API pārvaldniekā reģistrēta, savas vai citas iestādes, REST servisa (API) izsaukšanu, kas tiks reģistrēts API pārvaldniekā atbilstoši datu apmaiņu vadlīnijām [6], Lai no e-pakalpojuma izsauktu API pārvaldniekā reģistrētu servisu (API) e-pakalpojums nav jāreģistrē API pārvaldniekā (Store) kā klienta lietojums.

Lai izmantotu citu iestāžu servisus, kas ir aizsargāti ar atļaujām scopes, par to izmantošanu ir jāvienojas ar datu devējiem, atbilstoši datu apmaiņu vadlīnijām [6], datu devēji pēc tam piešķirs nepieciešamās tiesības.

> ***Ja tiek veidots SPA e-pakalpojums tad rekomendējams darbības ar biznesa datiem realizēt biznesa servisos (API), lai nodrošinātu maksimālu datu drošību, jo UI iestrādātā validācija var tikt apieta un BFF slānis tiek veidots kā publisks API un tajā tipiski nav iespējams veikt talona instrospekciju. Ja tas ir kritiski šāds scenārijs iepriekš ir jāsaskaņo ar VRAA – nepieciešams izveidot konkrētajam e-pakalpojumam atsevišķu klientu ar kuru varēs veikt talona introspekciju.***

Vienā e-pakalpojuma izpildes reizē (viens lietotājs atver un izpilda e-pakalpojumu) jāveido **viena** transakcija. E-pakalpojuma beigās transakcija jānoslēdz. Ja e-pakalpojums ir asinhrons (piemēram, darbība paredz pāreju uz biznesa servisu vai maksājumu moduli), transakcija jānoslēdz loģiskajā e-pakalpojuma noslēgumā. Transakcija ir vienots identifikators, kas ļauj atsekot viena e-pakalpojuma izpildes laikā veiktos pieprasījumus. Nepieciešams iestrādāt arī transakcijas noslēgšanu biznesa servisā izmantojot Transaction API [6], ja pakalpojuma darbība turpinās pēc e-pakalpojuma izpildes.

Lai veiktu API pārvaldniekā reģistrētu servisu izsaukumus, ir obligāti nepieciešams izveidot transakciju un nodefinēt izsaukumam atbilstošu pieturpunktu (milestone) un iekļaut to pieprasījumā.

Dažiem pieturpunktiem piesaista izpildes statusu "uzsākts" vai "pabeigts", lai uzkrātu korektu statistiku par to vai e-pakalpojumu izpilde no biznesa viedokļa tika pabeigta vai nē. Visi pieturpunkti tiks attēloti Latvija.lv portālā, to mērķis ir sniegt lietotājam priekštatu par e-pakalpojuma izpildes gaitu un konkrētas transakcijas izpildes progresu.

> ***Ja e-pakalpojuma arhitektūra paredz biznesa servisu izstrādi tad informāciju par lietotāju tajos ir jānolasa no saņemtā talona un obligāti jāpārliecinās ka pieprasītie dati pieder šim lietotājam. Šos servisus var izsaukt jebkurš autentificēts lietotājs apejot e-pakalpojuma UI un BFF tāpēc tajos jāparedz pilnvērtīga validācija, lai aizsargātos pret datu noplūdēm.***

Šī funkcionalitāte pieejama izmantojot e-pakalpojumu ietvara *LvpContext.Request* aizmugursistēmas metodes (skat. 7.2. nodaļu).

2.3. Elektronisko dokumentu krātuves (EDK) API

Elektronisko dokumentu krātuves (EDK) API nodrošina:

- Jauna dokumenta izveidošanu;
- Dokumenta īpašību izgūšanu;
- Dokumentu īpašību saraksta izgūšanu;
- Dokumenta īpašību aktualizēšanu;
- Dokumenta datnes izgūšanu;
- Dokumenta datnes aktualizēšanu;
- Dokumenta kopīgošanu;
- Dokumenta kopīgošanas pārtraukšanu;

Ja e-pakalpojuma biznesa process paredz datņu ielādi, ilglaicīgu uzglabāšanu vai pārsūtīšanu uz biznesa API (*back-end*), tad nepieciešams tās ielādēt elektronisko dokumentu krātuvē (EDK) un tālākām darbībām izmantot EDK dokumenta identifikatoru.

> ***EDK mapei, kas pieder e-pakalpojuma backend vai kādai ārējai sistēmai, ar kuru integrēsies e-pakalpojums jāpiešķir loma "AnyAuthenticatedEservice" ar "edk:getProperties" un "edk:createFolder" tiesībām! Šī loma ir nepieciešama, lai nodrošinātu iespēju kopīgot uz šo mapi e-pakalpojumā radītus dokumentus. Šīs darbības ir jānorāda e-pakalpojuma administratora dokumentā un izstrādes procesā jāprasa no VRAA.***

Tipisks EDK izmantošanas scenārijs - e-pakalpojumā izveidots dokuments jānodod uz biznesa servisu:

1. Ar VRAA saskaņo EDK hierarhiju kurā tiks uzglabāti biznesa vajadzībām izmantotie dokumenti. VRAA izveido saknes direktoriju ar atbilstošām tiesībām;
2. Lietotājs augšuplādē dokumentu vai tas tiek ģenerēts balstoties uz ievadītajiem datiem;
3. Izmantojot EDK API dokuments ar unikālu nosaukumu no e-pakalpojuma tiek saglabāts lietotāja hierarhijā, rezultātā API atgriež EDK dokumenta identifikatoru (URN);
4. Izveidoto dokumentu e-pakalpojums kopīgo izmantojot EDK API uz iepriekš saskaņoto biznesa datu hierarhiju (kopīgojot EDK dokumentu tā identifikators nemainās);
5. Tiek izsaukts biznesa serviss izmantojot Pieprasījumu API un body nodots EDK saglabātā dokumenta identifikators (URN);
6. Biznesa serviss apstrādā pieprasījumu:
    a. fiksē biznesa datu bāzē EDK dokumenta identifikatoru;
    b. veic citu servisu izsaukumus nododot tiem EDK dokumenta identifikatoru (URN);
    c. pēc EDK dokumenta identifikatora izgūst no EDK SOAP servisiem dokumenta datni vai metadatus izmantojot sertifikātu ar atbilstošām tiesībām, skatīt [5] un [11].

Tipisks EDK izmantošanas scenārijs – biznesa servisā izveidots dokuments jānodod uz e-pakalpojumu:

1. Ar VRAA saskaņo EDK hierarhiju kurā tiks uzglabāti biznesa valadzībām izmantotie dokumenti. VRAA izveido saknes direktoriju ar atbilstošām tiesībām;
2. No e-pakalpojuma tiek veikts biznesa servisa izsaukums izmantojot Pieprasījumu API;
3. Biznesa serviss izmantojot EDK SOAP servisu saglabā dokumentu iepriekš saskaņotajā biznesa datu hierarhijā, skatīt [5] un [11];

4. Biznesa serviss kopīgo dokumentu uz autentificētā lietotāja hierarhiju, skatīt [5] un [11] (kopīgojot EDK dokumentu tā identifikators nemainās);
5. Biznesa serviss atbildē uz e-pakalpojumu atgriež EDK dokumenta identifikatoru;
6. Izmantojot EDK API dokuments no e-pakalpojuma tiek lejupielādēts dokuments.

E-pakalpojumi var izmantot *EDK* servisa funkcionalitāti, izmantojot e-pakalpojumu ietvara *LvpContext.Edk* aizmugursistēmas metodes (skat. 7.3. nodaļu).

2.4. Notifikāciju API

Notifikāciju serviss nodrošina ziņojumu sūtīšanu latvija.lv lietotājiem (fiziskām, juridiskām un pilnvarotajām personām). Atbalstītie ziņojumu veidi ir:

- e-pasts (iespējams sūtīt gan uz konkrētu e-pasta adresi, gan uz lietotāja LVP profilā norādīto)
- KDV ziņojumi

E-pakalpojumi var izmantot Notifikāciju servisa funkcionalitāti, izmantojot e-pakalpojumu ietvara *LvpContext.Notification* aizmugursistēmas metodes (skat. 7.4. nodaļu).

Lai noformētu paziņojumus var izmantoto XSLT 1.0 transformācijas. Detalizētāku informācju skatīt dokumenta [3] nodaļā “4.6.3 Paziņojumi”.

2.5. Lietotāja profila API

Lietotāja profila API nodrošina iespēju no e-pakalpojuma nolasīt lietotāja profilā saglabātos datus, piemēram adresi un telefona numuru, un veikt to izmaiņas. Katram lietotāja tipam ir pieejams savs īpašību komplekts, detalizētāku informāciju par īpašībām un to nozīmi skatīt 1.tabulā. Pieeja pie profila datiem tiek kontrolēta balstoties uz konfigurācijas API profilePropertiesToRead un profilePropertiesToWrite struktūram. Lietotāju tipi:

- Iedzīvotājs – dati par fizisko personu tiek glabāti iedzīvotāja profilā, tas tiek identificēts pēc personas koda (PK). Šaja profilā tiek saglabāts, piemēram, iedzīvotāja personīgais e-pasts. Pieeja tiek atļauta tikai tām īpašībam, kas norādītas konfigurācijas API "profileType": "Person" struktūrā.
- Uzņēmuma pārstāvis – dati par uzņēmuma darbinieku (juridisku personu) no uzņēmumu reģistra, tas tiek identificēts pēc uzņēmuma reģistrācijas numura (UR) un personas koda (PK) kombinācijas (UR+PK). Šajā profilā tiek glabātas īpašības, kas attiecas uz konkrēto uzņēmuma darbinieku, piemēram, darbiniekam piešķirtais darba e-pasts. Pieeja tiek atļauta tikai tām īpašībām, kas norādītas konfigurācijas API "profileType": "Company" struktūrā.
- Pilnvarotais – dati par fiziskas personas deleģēto vai juridiskas personas pilnvaroto, tas tiek identificēts pēc pilnvaras devēja uzņēmuma reģistrācijas numura (DP) vai deleģētāja personas koda (DP) un pilnvaras saņēmēja personas koda (PK) kombinācijas (DP+PK). Pieeja tiek atļauta tikai tām īpašībām, kas norādītas konfigurācijas API "profileType": "Person" struktūrā, ja persona ir juridiskas personas pilnvarotais. Pieeja tiek atļauta tikai tām īpašībām, kas norādītas konfigurācijas API "profileType": "Person" struktūrā, ja persona ir fiziskas personas deleģētais.

1.tabula

Latvija.lv portāla lietotāju profilu īpašības (atkarībā no lietotāja tipa)

PROFILA ĪPAŠĪBA	APRAKSTS	ĪPAŠĪBAS ESAMĪBA PROFILĀ ATKARĪBĀ NO LIETOTĀJA TIPA		
		IEDZĪVOTĀJS (PK)	*PILNVAROTAIS (DP+PK)*	*UZŅĒMUMA PĀRSTĀVIS (UR+PK)*
Identifier	Vienotais identifikators	+	+	+
FirstName	Vārds	+	+	+
LastName	Uzvārds	+	+	+
Country	Valsts	+	+	+
Region	Rajons	+	+	+
City	Pilsēta	+	+	+
Village	Ciems	+	+	+
Street	Iela	+	+	+
HouseNumber	Mājas numurs	+	+	+
FlatNumber	Dzīvokļa numurs	+	+	+
PostIndex	Pasta indekss	+	+	+
FullAddress	Pilna adrese	+	+	+
Phone	Telefona numurs	+	+	+
Email	E-pasts	+	+	+
GetInfoOnEmail	Pazīme, kas norāda vai lietotājs ir piekritis saņemt paziņojumus uz Email īpašībā saglabāto e-pastu, neattiecas uz e-adreses ziņojumiem, tiem ir specifiska pazīme GetEaddresInfoOnMail.	+	+	+
BankAccountNumber	Bankas konta numurs	+	+	+
Territory	Teritorija no ATVK klasifikatora.	+	+	+
UseTerritory	Pazīme, kas norāda ka datu filtrēšanai jāizmanto lietotāja profilā norādītā teritorijā	+	+	+
ItemsPerPage	Vienā saraksta lapā attēlojamo ierakstu skaits.	+	+	+
CompanyRegistration-Number	Uzņēmuma reģistrācijas numurs		+	+
CompanyName	Uzņēmuma nosaukums		+	+
Position	Amats	+		

E-pakalpojumi var izmantot Lietotāja Profila servisa funkcionalitāti, izmantojot e-pakalpojumu ietvara *LvpContext.UserProfile* aizmugursistēmas metodes (skat. 7.5. nodaļu).

E-pakalpojumu katalogā katram e-pakalpojumam individuāli tiek definētas lasīšanas un rakstīšanas katrai īpašībai individuāli divām lietotāju grupām – Company (juridiskā persona, iestāžu un juridisko pilnvarotās personas) un Person (fiziskā persona).

> ***Primāri datus par autentificēto personu jāizgūst no talona, piemēram vārdu, uzvārdu, uzņēmuma nosaukumu, personas kodu vai reģistrācijas numuru, jo profila īpašību aizpildīšana ir neobligāta un nevar tikt nodrošināts, ka tās satur adekvātu informāciju, jo ir pieejamas rediģēšanai.***

2.6. Konfigurācijas API

Konfigurācijas API nodrošina e-pakalpojumu konfigurācijas izgūšanu no e-paklpojuma kataloga. E-pakalpojuma kataloga saturs tiek aizpildīts atbilstoši e-pakalpojuma izstrādātāja sagatavotajai dokumentācijai (Excel) – "E-pakalpojuma apraksta šablons", pieejamas https://viss.gov.lv/lv/Informacijai/Dokumentacija/Koplietosanas_komponentes/EPAK_izstrades_izpildes_vide.

Var izmantot Konfigurācijas servisa funkcionalitāti, izmantojot e-pakalpojumu ietvara *LvpContext.Configuration* aizmugursistēmas metodes (skat. 7.6. nodaļu).

2.7. Maksājumu moduļa API

Serviss nodrošina e-pakalpojumu izstrādātājiem veidot maksas pakalpojumus, veicot apmaksu pakalpojuma laikā vai sagatavojot rēķinu un nodrošinot norēķināšanos ar pēcapmaksu. Lai integrētos ar maksājumu moduli skatīt [7] dokumentu.

2.8. Adrešu meklēšanas komponente (AMK)

Adrešu meklēšanas komponente ir paredzēta adrešu meklēšanai Adrešu reģistra datu bāzē. Adrešu meklēšanas komponentes grafiskie resursi izvietoti iekš Assets, bet tā darbības nodrošināšanai nepieciešamais API izmitināts VRAA infrastruktūrā. AMK komponentes izmantošanas piemēri ir aprakstīti React, HTML storybook un MVC helpers dokumentācijā.

- React: https://eservices-test.vraa.gov.lv/EservicePlatform.Controls.React/?path=/story/components-addressfinder--address-finder
  - HTML: https://eservices-test.vraa.gov.lv/EservicePlatform.Controls.Html/?path=/story/components-addressfinder--address-finder
  - MVC helpers: https://eservices-test.vraa.gov.lv/EservicePlatform.Controls.Mvc/home/helper?name=AddressFinder

Papildus informāciju skatīt AMK integrācijas instrukcijā [8].

2.9. Kļūdu pieteikšanas API

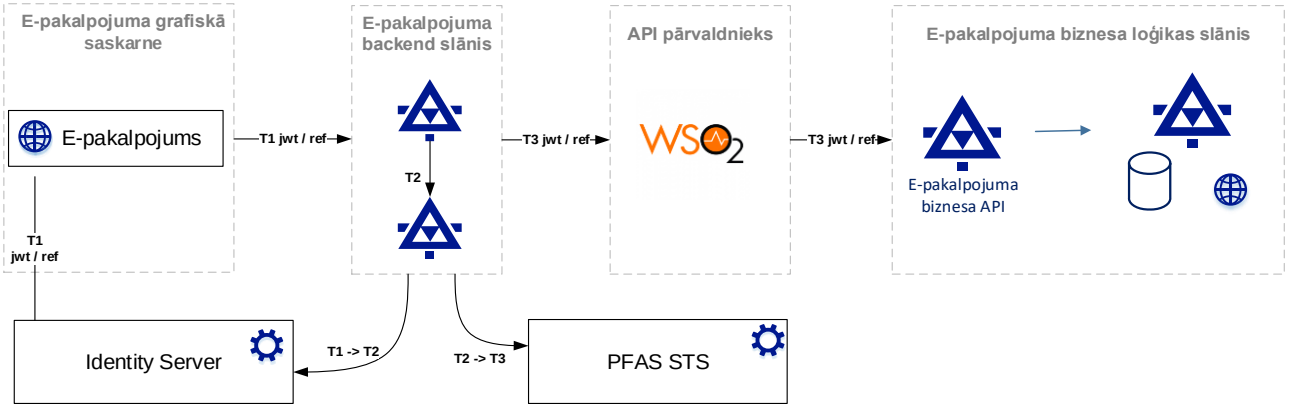
Kļūdu pieteikšanas API paredzēts e-pakalpojumu integrācijas nodrošināšanai ar kļūdu pieteikšanas risinājumu. Tas ļauj no e-pakalpojuma iesniegt pieteikumu JSON formātā, kuru tālāk saformē kā IvisRequest dokumentu un nosūta uz pieprasījumu servisu. Lai integrētos ar kļūdu pieteikšanas api skatīt 7.8 nodaļu.

2.10. Epakalpojumu biznesa API

Izstrādājot e-pakalpojumu var būt nepieciešams integrēties ar esošu risinājumu vai radīt jaunu biznesa loģikas slāni. Lai nodrošinātu integrāciju starp biznesa loģikas slāni un e-pakalpojumu nepieciešams realizēt REST servisu atbilstoši datu apmaiņu vadlīnijām (skatīt [6]).

Lai nodrošinātu ka biznesa servisi (API) ir pieejami tikai konkrētiem e-pakalpojumiem tos nepieciešams reģistrēt API pārvaldniekā un pieeju ierobežot definējot scopes katrai no eksponētajām metodēm.

Lai ierobežotu pieeju pie datiem ir iespējams e-pakalpojuma biznesa API līmenī pārbaudīt ziņojuma header iekļauto IDS talonu un atļaut piekļūt tikai pie lietotājam piederošajiem datiem. Detalizētāku informāciju par API izstrādi skatīt [6] dokumentā.



2.attēls. E-pakalpojuma risinājuma arhitektūra

> ***Ja tiek veidots SPA e-pakalpojums, tad rekomendējams darbības ar biznesa datiem realizēt biznesa servisos, lai nodrošinātu maksimālu datu drošību, jo UI iestrādātā validācija var tikt apieta un BFF slānis tiek veidots kā publisks API un tajā tipiski nav iespējams veikt talona instospekciju.***
>
> ***E-pakalpojuma biznesa servisos informāciju par lietotāju ir jānolasa no saņemtā PFAS talona, ja nepieciešams var tikt izsaukts PFAS introspect, lai pārbaudītu talona derīgumu vai iegūtu tā saturu. Biznesa servisus var izsaukt jebkurš autentificēts lietotājs apejot e-pakalpojuma UI un BFF tāpēc tajos jāparedz pilnvērtīga validācija, lai aizsargātos pret datu noplūdēm.***

2.11.Assets

E-pakalpojumu standarta resursi (attēli, JS un CSS datnes) tiek piegādāti, izmantojot atsevišķu tīmekļa lietojumu (Assets).

Assets paredzēts e-pakalpojumu koplietojamo (atkārtoti izmantojamo) resursu glabāšanai. Šāds risinājums samazina koda dublēšanu, ļauj veikt resursu optimizāciju (*minification*), kā arī uzlabo tīmekļa pārlūka kešdarbes iespējas.

E-pakalpojumu specifiskos resursus paredzēts glabāt pašā e-pakalpojumā.

Assets satur vairākas resursu versijas. Piegādātās Asset versijas netiek labotas, lai izvairītos no problēmām ar pārlūku kešdarbi. Nepieciešamie labojumi un uzlabojumi tiek veikti jaunā (nākošajā) versijā. Izstrādājot e-pakalpojumu, jāizmanto pēdējā versija.

Asseti sastāv no trim galvenajām daļām:

- globālie resursi attēli, js, css kuru izmanto e-pakalpojumi, react mvc html komponentes, utt
- react komponentes un tikai to darbībai nepieciešamie resursi (attēli, js, css),
- react.js un react-dom.js bibliotēkas,
- kā arī citu komponenšu resursi

Resursiem ir liegta direktoriju pārlasīšana, bet prasot konkrētu resursu tas tiek atgriezts. Resursu var izgūt veicot pieprasījumu uz to atrašanās vietu pēc šāda parauga:

<domēns>/<versija>/<komponente>/<ceļš līdz resursam>

- Domēns – domēnvārds, kur atrodas resursi
- Versija – assetu versija, kurā atrodams resurss
- Komponente – Sistēmas komponente, kurai resurss paredzēts. Dažkārt komponentes iedalās sīkāk piemēram SDK iedalās HTMLSDK un ReactSDK. Tad būtu nepieciešams norādīt ceļu SDK/HTMLSDK/…
- Ceļš līdz resursam – ceļš, kuru norāda komponentes izstrādātājs, piemēram images/logo.png

Piemērs. https://eservices-test.vraa.gov.lv/EservicePlatform.Assets/v2.1.0/Global/images/logo.png

Piemērā tiek izgūts globālais assets – logo, kas ir bilde zem globālajiem assetiem 2.1.0 versijā.

Šobrīd pieejamā assetu struktūra:


Amk
Content
img
Vraa.AddressFinder.css
Scripts
v4
Vraa.AddressFinder.js
Vraa.AddressFinder.js
DevExtreme
css
dx.common.css
dx.generic.latvija.css
dx.generic.latvija.json
Global
images
assets
actuality-mockup.jpg
e-adrese.png
e-pak-mockup.png
e-pak-notebook.png
icons
logos
auth
cef-lv@2x.png
eraf@2x.png
eugo-logo-rgb-60@2x.png
logo.png
logo.svg
orange-bitmap.png
orange-bitmap-min.png
red-bitmap.png
red-bitmap-min.png
js
react
@16.14.0
react.development.js
react.production.min.js
react-dom.development.js
react-dom.production.min.js
daterangepicker@3.1.0.min.js
jodit@3.4.23.min.js
jquery@3.5.1.min.js
moment@2.27.0.min.js
SDK
HTMLSDK
ReactSDK

2.12.Autentifikācija

Autentifikācijai tiek izmantots SSO. Ja lietotājs ir ielogojies Latvija.lv portālā, atverot e-pakalpojumu, notiek automātiska autentifikācija arī e-pakalpojuma lietojumā.

Savukārt, ja lietotājs nav ielogojies LVP, pie e-pakalpojuma atvēršanas caur konfigurācijas API tiek noskaidrots, vai e-pakalpojums ir publisks. Ja e-pakalpojums nav publisks, lietotājam tiek attēlots autentifikācijas logs un e-pakalpojums netiek atvērts.

Ja e-pakalpojums ir anonīms, tad automātiskā autentifikācija nenotiek, ja ir nepieciešams uzsākt anonīmo e-pakalpojumu ar autentifikāciju, pieprasījumā ir jāpadod papildus parametrs – authenticated=1.

E-pakalpojumos talons vienmēr tiek izsniegts Konteksta API izsaukšanai, audience atribūts “aud” satur Konteksta API atbilstošu identifikatoru. Pat ja risinājums paredz introspekciju BFF līmenī (Iepriekš jāsaskaņo ar VRAA).

Autentifikāciju SPA variantā nodrošina npm pakotne, kura ir aprakstīta 6. punktā, izstrādātājam ir jānodrošina tikai autorizācijas konfigurācijas parametri.

MPA autorizācijas kontrolierus nodrošina 6. punktā aprakstītā nuget pakotne. Autentifikācijas konfigurācijas parametrus izstrādātājam ir jānorāda projekta appsettings.json failā. Autentifikācijas konfigurāciju startup.cs failam ir jākopē no e-pakalpojumu piemēriem.

Visos e-pakalpojumos tiek izmantota vienota pieeja veidojot saites autentifikācijas nodrošināšanai.

SPA e-pakalpojumiem atļauts izmantot šādas adreses:

- login: {e-pakalpojuma adrese}/signin-oidc.
- renew: {e-pakalpojuma adrese}/silent-renew-callback.
- logout: {e-pakalpojuma adrese}/signout-callback-oidc.

MPA e-pakalpojumiem atļauts izmantot šādas adreses:

- login: {e-pakalpojuma adrese}/signin-oidc
- logout: {e-pakalpojuma adrese}/signout-callback-oidc

> ***Primāri datus par autentificēto personu jāizgūst no talona nevis lietotāja profila, piemēram vārdu, uzvārdu, uzņēmuma nosaukumu, personas kodu vai reģistrācijas numuru, jo profila īpašību aizpildīšana ir neobligāta un nevar tikt nodrošināts ka tās satur adekvātu informāciju, jo ir pieejamas rediģēšanai.***

> ***E-pakalpojuma biznesa servisos informāciju par lietotāju ir jānolasa no saņemtā PFAS talona, ja nepieciešams var tikt izsaukts PFAS introspect, lai pārbaudītu talona derīgumu vai iegūtu tā saturu. Biznesa servisus var izsaukt jebkurš autentificēts lietotājs apejot e-pakalpojuma UI un BFF tāpēc tajos jāparedz pilnvērtīga validācija, lai aizsargātos pret datu noplūdēm.***

2.13.Pilnvarotās personas

E-pakalpojumu ietvars nodrošina nepieciešamās pārbaudes un paziņojumu attēlošanu darbam ar pilnvarotajām personām:

1. Tiek pārbaudīts vai e-pakalpojumu ir atļauts izmantot pilnvarotajiem un deleģētajiem balstoties uz konfigurācijas API pazīmi isForDelegatedPersons – true atļauts, bet false aizliegts;
2. Tiek pārbaudīts vai lietotājs ir autentificējies ar atļautu autentifikācijas provaideri. Fiziskas personas deleģētā gadījumā tiek pārbaudīts balstoties uz konfigurācijas API pazīmi

inhabitantIdentityProviders. Juridiskas personas pilnvarotā gadījumā tiek pārbaudīts balstoties uz konfigurācijas API pazīmi legalEntityIdentityProviders.

3. Tiek pārbaudīts vai ir derīga pilnvara un tajā atļauta konkrēta e-pakalpojuma izmantošana izmantojot Access API, skatīt 7.11 nodaļu. Pilnvarošanas procesi pašreiz neparedz iespēju dot tiesības izmantot e-pakalpojumu citas fiziskas personas vārdā.

2.14.Integrācija ar Latvija.lv un citiem portāliem

E-pakalpojumi ir integrēti Latvija.lv portālā, un tas nodrošina nepieciešamo datu nodošanu uz e-pakalpojumu:

- Pārvirza uz e-pakalpojuma sākuma soli nododot portālā izvēlēto valodu - {e-pakalpojuma adrese}/lv/eservice/start.
- Nodod informāciju par portāla izvēlēto teksta izmēru.
- Nodod informāciju par porālā izvēlēto vājredzīgo režīmu.
- Nodod informāciju par portālā autentificēto lietotāju.
- Nodod papildus parametrus, ja tādi ir atļauti.

Tiek nodrošināta pāreja no ārēja portāla uz e-pakalpojumu izmantojot saiti: https://latvija.lv/Epakalpojumi/{e-paklpojuma numurs, piemēram EP200}. Tā nodrošina ka uz pakalpojumu tiks nodoti lietotāja pārlūka cookie saglabātie portāla parametri vai izmantotas noklusētās vērtības:

- Vājredzīgo režīmam;
- Teksta izmēram;

Lai nodotu noteiktu valodu jāizmanto saite https://latvija.lv/{valoda}/Epakalpojumi/{e-paklpojuma numurs, piemēram EP200}, kur {valoda} vietā jānorāda:

- lv – latviešu valodai (pēc noklusējuma var neaizpildīt).
- ru – krievu valodai.
- en – angļu valodai.

Lai nodotu papildus parametru jāizmanto saite https://latvija.lv/Epakalpojumi/{e-paklpojuma numurs, piemēram EP200}?{parametra nosaukums}={parametra vērtība}. Iespējams nodot vairākus parametrus atdalot tos ar & simbolu, piemēram, https://latvija.lv/Epakalpojumi/{e-paklpojuma numurs, piemēram EP200}?{1. parametra nosaukums}={1.parametra vērtība}&{2. parametra nosaukums}={2.parametra vērtība}.

Lai pārietu uz e-pakalpojuma apraksta lapu jāizmato šāda saite https://latvija.lv/{valoda}/Epakalpojumi/{e-pakalpojuma numurs, piemēram EP200}/Apraksts. Tā nodrošina e-pakalpojuma apraksta un lietošanas noteikumu attēlošanu. Izmantojot šo saiti nevar nodot uz e-pakalpojumu papildus parametrus, bet tā nodrošina ka uz pakalpojumu tiks nodoti lietotāja pārlūka cookie saglabātie portāla parametri vai izmantotas noklusētās vērtības:

- Vājredzīgo režīmam;
- Teksta izmēram;

3. Izstrādes vides sagatavošana

Šajā sadaļā aprakstīts, kas nepieciešams un kas jādara, lai sagatavotu izstrādes vidi priekš e-pakalpojuma izstrādes. Pirms e-pakalpojuma izstrādes no VRAA jāpieprasa e-pakalpojuma identifikators (URN) un IDS klienta dati autentifikācijas nodrošināšanai e-pakalpojumā.

3.1. Izstrādes vides prasības

Lai uzstādītu izstrādes vidi ar docker, tad nepieciešams uzstādīt šādu programmatūru:

- Git
- Docker
- Docker-compose

Lai uzstādītu izstrādes vidi bez docker, tad nepieciešams uzstādīt šādu programmatūru:

- Git;
- Node;
- NuGet (https://www.nuget.org/downloads);
- NPM;
- .Net Core 3.1.

3.2. Izstrādei nepieciešamie resursi

Resursi pieejami tikai pēc pieprasījuma, no VRAA:

- React un MVC e-pakalpojumu piemēru pirmkodi pieejami Git repozitorijos https://git.vraa.gov.lv/lvp/eserviceplatform.examples/react un https://git.vraa.gov.lv/lvp/eserviceplatform.examples/mvc. Sīkāki apraksti katram piemēram atrodas 9.nodaļā. Pieejami šādi piemēri:
  - Komponenšu demonstrācijai (ComplexUI);
  - Servisu izsaukumu demonstrācijai (SercieIntegration);
  - Pilnvaroto personu apstrādei (EP500);
- React un MVC e-pakalpojumu šabloni (Template) pieejami Git repozitorijos https://git.vraa.gov.lv/lvp/eserviceplatform.examples/react un https://git.vraa.gov.lv/lvp/eserviceplatform.examples/mvc.
- Docker un Docker-compose konfigurāciju, kuru iespējams izmantot lokālas vides uzstādīšanai. Docker-compose konfigurācijā ir aprakstīti visi piemēri un tos iespējams darbināt vienlaicīgi. Docker un docker-compose datnes atrodās attiecīgo piemēru direktorijās. Piemēri pieejami Git repozitorijos https://git.vraa.gov.lv/lvp/eserviceplatform.examples/react un https://git.vraa.gov.lv/lvp/eserviceplatform.examples/mvc.
- Izstrādei nepieciešamās bibliotēkas pieejamas no VRAA nuget - https://nexus.vraa.gov.lv/repository/eservices-nuget, izmantot jaunākās versijas:
  - Abc.Analytics.Serilog – žurnalēšanai izmantojamā bibliotēka, piemēram:

```
Install-Package Abc.Analytics.Serilog -Version 0.1.1
```

  - Abc.Analytics.Serilog.AspNetCore – žurnalēšanai izmantojamā bibliotēka, piemēram:

```
Install-Package Abc.Analytics.Serilog.AspNetCore -Version 0.1.1
```

- Lvp.EservicePlatform.Backend.Essentials.EserviceCore – ietvara funkcionalitātes bibliotēka, piemēram:

```
Install-Package Lvp.EservicePlatform.Backend.Essentials.EserviceCore -Version 1.0.46
```

- Lvp.EservicePlatform.Controls.Mvc – MVC helper bibliotēka, piemēram:

```
Install-Package Lvp.EservicePlatform.Controls.Mvc -Version 1.0.23
```

- Pieeja pie IsolatedContext docker lvp/eserviceplatform/backend/isolatedcontextapi tiek izsniegta pēc pieprasījuma VRAA un ir pieejams VRAA Nexus, piemēram (izmantot jaunāko versiju) https://nexus.vraa.gov.lv/repository/docker-private/v2/lvp/eserviceplatform/backend/isolatedcontextapi/manifests/1.0.4 .
- Pieeja pie izstrādei nepieciešamajām node pakotnēm tiek izsniegta pēc pieprasījuma VRAA un ir pieejamas VRAA Nexus, izmantot jaunākās versijas, piemēram:
  - https://nexus.vraa.gov.lv/repository/eservices-npm/@eserviceplatform/frontend-react/-/frontend-react-1.0.39.tgz
  - https://nexus.vraa.gov.lv/repository/eservices-npm/@eserviceplatform/controls-react/-/controls-react-1.0.53.tgz

MVC piemēri ir veidoti MPA (*Multi Page Application*) režīmā.

React piemēri ir veidoti SPA (*Single Page Application*) režīmā, katram piemēram ir grafiskās saskarnes daļa (webapp) un React servera daļa - (bff -*Backend For Frontend*), kura nodrošina starpslāni starp grafisko saskarni un e-pakalpojuma biznesa servisiem, detalizētu informāciju par komponenšu nosaukumu veidošanu skatīt dokumentā [10].

3.3. Izstrādes vides uzstādīšana

3.3.1. MPA vides uzstādīšana

Izstrādes vides uzstādīšana bez docker:

1. Kopēt datni appsettings.Example.json no saknes direktorijas uz projekta direktoriju.
2. Pārsaukt kopēto datni par appsettings.json
3. Aizpildīt konfigurācijas parametrus pārsauktajā datnē:

```
"oidc": { // open id connect autorizācijai nepieciešamā informācija
    // epakalpojuma klienta id
    "clientId": "urn:oauth2:test-vraa:complexui:mvc",
    // epakalpojuma klienta noslēpums
    "clientSecret":
"14DA6570F55272A1C71775603C6A7F178FAA9EDF8D8F1A93E1FBDEA41FF78E77",
    // autentifikācijas ievadformas adrese
    "authorizationEndpoint" : "https://epakvisstv.vraa.gov.lv/IdentityServer"
```

```
},
"Config": { // sistēmas parametri
    // assets adrese ar versijas vietturi
    "AssetsUri": "https://eservices-
test.vraa.gov.lv/EservicePlatform.Assets/{{version}}",
    // assetu versija - nepiecieešams norādīt tikai lokālajai izstrrādei
    "AssetsUriVersion": "",
    // sistēmas bāzes adrese
    "BaseUri": "https://localhost:5001",
    // contextApi publiska adrese
    "PublicContextApiUri": "https://eservices-
test.vraa.gov.lv/EservicePlatform.ContextApi",
    // iekšējā(k8s) ContextApi adresee. Prriekš lokālās izstrādes publiskā
    "ContextApiUri": "https://eservices-
test.vraa.gov.lv/EservicePlatform.ContextApi",
    // epakalpojuma URN
    "EServiceUrn": "URN:IVIS:100001:EP-EXAMPLECOMPLEXUIMVC-V1-0",
    // navigācijas servisa adrese
    "NavigationServiceUri": "https://eservices-
test.vraa.gov.lv/EservicePlatform.NavigationApi",
    // globālo tulkojumu resursa adrese
    "TranslationGlobalUri":
"https://epakvisstv.vraa.gov.lv/Lvp.EservicePlatform.Resources/Global/global.yaml
",
    // tulkojumu resursa adrese
    "TranslationUri":
"https://epakvisstv.vraa.gov.lv/Lvp.EservicePlatform.Resources/MVCExamples/comple
xui.yaml",
    // meklētāja lauka mērķadrese
    "SearchUri": "https://lvptest.vraa.gov.lv/{language}/Meklesana",
    // pazīmē par pielāgotas izstrrādes sākumlapu
    "OverloadIndex": "true",
    // sertifikāts priekš maksājumu servisa, ja netiek izmantots, tad nav
nepieciešams
    "PaymentCertificateBase64": "<cert>",
    // Adrese uz pildīto epakalpojumu sarakstu
    "ExecutedEserviceUrl":
"https://lvptest.vraa.gov.lv/{language}/KDV/E_pakalpojumu_saraksts ",
    // Minūtes pēc cik ilgas bezdarbības lietotāju izlogo no sistēmas
    "IdleLogoutTimeoutMinutes": 10,
    // Massīvs ar adresēm
    "Breadcrumbs" : [
     {
        "lv":"Sākums",
        "ru":"Начало",
        "en":"Home",
```

```
        "link":"https://lvptest.vraa.gov.lv/{language}"
     },
     {
        "lv":""E-pakalpojumi"
        "ru":""Э-услуги"
        "en":"E-services"
        "link":"https://lvptest.vraa.gov.lv/{language}/Epakalpojumi”
     },
     {
        "lv":"{eserviceName}",
        "en":"{eserviceName}",
        "ru":"{eserviceName}",
        "link":"https://lvptest.vraa.gov.lv/{language}/Epakalpojumi/{eserviceId}”
     },
     // Kriptēšanas atslēga prieks autorizācijas no portāla
     "QueryEncryptionKey": "<key>"
    ]
},
```

4. Konfigurēt izvēlētā projekta datni ./properties/launchsettings.json atbilstoši izstrādes videi. Projekts jādarbina adresē, kura norādīta sistēmas parametros kā BaseUri.
5. Konfigurēt izvēlētā projekta datni nuget.config

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <packageSources>
    <add key="VRAA" value="https://nexus.vraa.gov.lv/repository/eservices-nuget/"
/>
  </packageSources>
  <packageSourceCredentials>
      <VRAA>
          <add key=”Username” value=”nexus-username”>
          <add key=”ClearTextPassword” value=”nexus-password”>
      </VRAA>
  </packageSourceCredentials>
</configuration>
```

6. Darbināt projektu izvēlētajā vidē. Šobrīd zināmās izstrādes vides uz kurām darbināts projekts –
    a. Jetbrains Rider
    b. VS Code
    c. Visual studio

Izstrādes vides uzstādīšana ar docker:

1. Pārsaukt saknes direktorijā esošo datni appsettings.Example.json par appsettings.json
2. Aizpildīt konfigurācijas parametrus appsettings.json datnē (skat attēlu augstāk redzamajā aprakstā “uzstādīšana bez docker” zem 3. punkta)
3. Pārsaukt saknes direktorijā esošo .env.examples datni par .env
4. Aizpildīt konfigurācijas parametrus .env datnē

```
# vide development|production
ENVIRONMENT=development
# publiskās adreses celš
PUBLIC_URL=http://localhost
# ports kurā pieejams complex ui piemērs
PUBLIC_PORT_COMPLEX_UI=5013
# ports kurā pieejams EP500 piemērs
PUBLIC_PORT_EP500=5015
# ports kurā pieejama servisa integrācija piemērs
PUBLIC_PORT_SERVICE_INTEGRATION=5014
# celš uz konfigurācijas failu, kurš tiks ievietots docker konteinerī
APPSETTINGS_SRC_PATH=./appsettings.json
```

5. No Saknes direktorijas izpildīt komandu(servisa nosaukumu iespējams atrast docker-compose.yml datnē zem mainīgā services) - *docker-compose up --build <servisa nosaukums>*
6. Ja nepieciešams darbināt visus projektus vienlaicīgi, tiem augstāk redzamajā .env datnes attēlā ir nepieciešams norādīt atšķirīgu portus no kuriem tie darbojās, kā arī katram norādīt savu appsettings failu, dublējot un piekoriģējot šī apraksta 3. datni. Lai docker-compose katram projektam piemērotu savu appsettings, tad docker-compose.yml failā katram servisam zem volume parametriem norādītā mainīgā ${APPSETTINGS_SRC_PATH} vietā jānorāda adrese uz nepieciešamo appsettings failu.

```
services:
  lvp-mvc-complex-ui:
    container_name: 'Lvp.EservicePlatform.Examples.Mvc.ComplexUI'
    image: examples/mvc/complex-ui
    environment:
      env_file: .env
      PUBLIC_URL: ${PUBLIC_URL}:${PUBLIC_PORT_COMPLEX_UI}
    ports:
      - '${PUBLIC_PORT_COMPLEX_UI}:80'
    build:
      context: .
      dockerfile: ./src/Lvp.EservicePlatform.Examples.Mvc.ComplexUI/Dockerfile
    volumes:
      - '${APPSETTINGS_SRC_PATH}:/app/appsettings.json'
```

3.3.2. SPA vides uzstādīšana

Izstrādes vides uzstādīšana bez docker:

1. Kopēt datni .env-react.example no projekta saknes direktorijas uz izvēlēto react projektu(kuri atrodas zem direktorijas webapp) un pārsaukt par .env

2. Aizpildīt .env datnes vērtības ar reālajām vērtībām

```
#---------------------
# JS izejas faila konfigurācija.
# Šos parametrus izmanto npm priekš izejas faila izveides šos parametrus jāizmanto
#---------------------
# Pilna saite uz vietni
PUBLIC_URL={{IMAGE_PUBLIC_URL}}
# vides parametrs development|production
ENVIRONMENT={{IMAGE_ENVIRONMENT}}
# Ports tiks izvietota lietotne
PUBLIC_PORT={{IMAGE_PUBLIC_PORT}}
# Autorizācijas parametri
# Saite uz Autorizācijas ievadformu
AUTH_AUTHORITY_API_ENDPOINT={{IMAGE_AUTH_AUTHORITY_API_ENDPOINT}}
# Epkalpojuma autorizācijas klienta ID
AUTH_CLIENT_ID={{IMAGE_AUTH_CLIENT_ID}}
# Epkalpojuma autorizācijas klienta noslēpums
AUTH_CLIENT_SECRET={{IMAGE_AUTH_CLIENT_SECRET}}
# Epkalpojuma autorizācijas tvērums
AUTH_SCOPE={{IMAGE_AUTH_SCOPE}}
# Saite uz epakalpojuma BFF(backend-for-frontend)
ESERVICE_API_ENDPOINT_URL={{IMAGE_ESERVICE_API_ENDPOINT_URL}}
# epakalpojuma identifikators
ESERVICE_URN={{IMAGE_ESERVICE_URN}}
# Saite uz NavigationBar servisu
NAVIGATION_API_ENDPOINT_URL={{IMAGE_NAVIGATION_API_ENDPOINT_URL}}
# Saite uz Assetu glabātuvi
ASSETS_CDN_URL={{IMAGE_ASSETS_CDN_URL}}
# Saite contextApi servisu
CONTEXT_API_ENDPOINT_URL={{IMAGE_CONTEXT_API_ENDPOINT_URL}}
# Saite uz meklēšanas servisi
SEARCH_ENDPOINT_URL={{IMAGE_SEARCH_ENDPOINT_URL}}
# Saite uz latvija.lv portālu
EXECUTED_ESERVICE_URL={{IMAGE_EXECUTED_ESERVICE_URL}}
# laiks minūtēs pēc cik ilgas bezdarbības  lietotāja tiks veikta pārbaude par aktivitāti
IDLE_LOGOUT_TIMEOUT_MINUTES={{IMAGE_IDLE_LOGOUT_TIMEOUT_MINUTES}}
# portala sākumlapas adrese
PORTAL_HOME_URL={{IMAGE_PORTAL_HOME_URL}}
# portala profila adrese
PORTAL_PROFILE_URL={{IMAGE_PORTAL_PROFILE_URL}}
# breadcrumb ieraksti
```

```
ESERVICE_BREADCRUMB={{IMAGE_ESERVICE_BREADCRUMB}}
# izstrādes vidēlai padotu sertifikātus ir jānorāda to atrašanās direktorija
#CERT_PATH=/etc/ssl/localcerts
```

3. Kopēt datni apssettings.Example.json uz izvēlēto SPA projekta starpslāni kuri ir atrodami zem direktorijas api/src.
4. Izvēlētā projekta direktorijā konfigurēt datni .npmrc

```
@eserviceplatform:registry=https://nexus.vraa.gov.lv/repository/eservices-npm
update-notifier=false
_auth=<base64encode(nexus-username:nexus-password)>
```

5. Izvēlētā projekta direktorijā izpildīt komandu *npm install*
6. Izvēlētā projekta direktorijā izpildīt komandu *npm run start*
7. Pārsaukt kopēto datni par appsettings.json
8. Aizpildīt appsettings.json datnes konfigurācijas vērtības.

```
"Config": {
  "AssetsUri":"https://eservices-test.vraa.lv/EservicePlatform.Assets",
  "BaseUri": "https://localhost:5002",
  "PublicContextApiUri": "https://eservices-
test.vraa.lv/EservicePlatform.ContextApi",
  "ContextApiUri": "https:// eservices-test.vraa.lv/EservicePlatform.ContextApi",
  "EServiceUrn": "URN:IVIS:100001:EP-EXAMPLECOMPLEXUIMVC-V1-0",
  "NavigationServiceUri": "https:// eservices-test.vraa.lv
/EservicePlatform.NavigationApi",
  "TranslationGlobalUri": "Resources/global.yaml",
  "TranslationUri": "Resources/complexui.yaml",
  "SearchUri": "https://www.latvija.lv/lv/Meklesana"
  // sertifikāts priekš maksājumu servisa, ja netiek izmantots, tad nav
nepieciešams
   "PaymentCertificateBase64": "<cert>",
  // Kriptēšanas atslēga prieks autorizācijas no portāla
   "QueryEncryptionKey": "<key>"
  },
```

9. Konfigurēt izvēlētā projekta datni ./properties/launchsettings.json atbilstoši izstrādes videi. Projekts jādarbina adresē, kura norādīta sistēmas parametros kā BaseUri.
10. Darbināt projektu izvēlētajā vidē. Šobrīd zināmās izstrādes vides uz kurām darbināts projekts
    a. Jetbrains Rider
    b. VS Code
    c. Visual studio

Izstrādes vides uzstādīšana ar docker:

1. Pārsaukt saknes direktorijā esošo datni appsettings.Example.json par appsettings.json

2. Aizpildīt konfigurācijas parametrus appsettings.json datnē (skat attēlu pie 7. punkta uzstādīšanas bez docker)
3. Pārsaukt saknes direktorijā esošo .env.examples datni par .env
4. Aizpildīt konfigurācijas parametrus .env datnē

```
# publiskā bāzes adrese visiem projektiem
PUBLIC_URL=https://localhost
# ceļš uz kopīgo appsetings konfigurācijas datni
APPSETTINGS_SRC_PATH=./appsettings.json
# complex ui izmantotie porti
PUBLIC_PORT_COMPLEX_UI_BFF=5001
PUBLIC_PORT_COMPLEX_UI_REACT=5002
# ep500 izmantotie porti
PUBLIC_PORT_EP500_BFF=5005
PUBLIC_PORT_EP500_REACT=5006
# service integration izmantotie porti
PUBLIC_PORT_SERVICE_INTEGRATION_BFF=5007
PUBLIC_PORT_SERVICE_INTEGRATION_REACT=5008
#----------------------
# Docker konteinerī padodamās vērtības
# šīs vērtības tiek izmnantotas, lai JS izejas failā aizvietotu
# vietturus ar reālajām vērtībām
#----------------------
# kopīgie parametri
# attēla vide
IMAGE_ENVIRONMENT=development
# Saite uz Autorizācijas ievadformu
IMAGE_AUTH_AUTHORITY_API_ENDPOINT=https://eservices-
test.vraa.lv/Portal.IdentityServer
# Saite uz epakalpojuma BFF(backend-for-frontend)
IMAGE_ESERVICE_API_ENDPOINT_URL=https://localhost:5002/api
# Saite uz meklēšanas servisi
IMAGE_SEARCH_ENDPOINT_URL=https://www.latvija.lv/lv/Meklesana
# Saite uz latvija.lv portālu
IMAGE_EXECUTED_ESERVICE_URL=https://www.latvija.lv
# Saite uz portala sākumlapu
IMAGE_PORTAL_HOME_URL=https://www.latvija.lv
# Saite uz portala lietotāja profilu
IMAGE_PORTAL_PROFILE_URL=https://www.latvija.lv
# Saite uz NavigationBar servisu
IMAGE_NAVIGATION_API_ENDPOINT_URL=https://eservices-
test.vraa.lv/EservicePlatform.NavigationApi
# Saite uz Assetu glabātuvi
IMAGE_ASSETS_CDN_URL=https://eservices-test.vraa.lv/EservicePlatform.Assets
```

```
# Saite contextApi servisu
IMAGE_CONTEXT_API_ENDPOINT_URL=https://eservices-
test.vraa.lv/EservicePlatform.ContextApi
# laiks minūtēs pēc cik ilgas bezdarbības  lietotāja tiks veikta pārbaude par
aktivitāti
IMAGE_IDLE_LOGOUT_TIMEOUT_MINUTES=10
# breadcrumb ieraksti
IMAGE_ESERVICE_BREADCRUMB=[{"lv":"Sākums","ru":"Главная     Страница","en":"Home",
"link":"{baseAddress}/{language}"}]
#complex ui parametri
COMPLEXUI_IMAGE_PUBLIC_URL=http://localhost:5002
COMPLEXUI_IMAGE_AUTH_CLIENT_ID=urn:oauth2:cs:complexui:react
COMPLEXUI_IMAGE_AUTH_CLIENT_SECRET=8C4E6AFC3B4D278521C2D8524A9B285C5B54C28290C48D
4D170F06931FF12998
COMPLEXUI_IMAGE_AUTH_SCOPE="openid profile epak context_api"
COMPLEXUI_IMAGE_ESERVICE_API_ENDPOINT_URL=https://eservices-
test.vraa.lv/EservicePlatform.Examples.React.ComplexUIApi/api
COMPLEXUI_IMAGE_ESERVICE_URN=URN:IVIS:100001:EP-EXAMPLECOMPLEXUIMVC-V1-0
#ep500 parametri
EP500_IMAGE_PUBLIC_URL=http://localhost:5004
EP500_IMAGE_AUTH_CLIENT_ID=urn:oauth2:cs:complexui:react
EP500_IMAGE_AUTH_CLIENT_SECRET=8C4E6AFC3B4D278521C2D8524A9B285C5B54C28290C48D4D17
0F06931FF12998
EP500_IMAGE_AUTH_SCOPE="openid profile epak context_api"
EP500_IMAGE_ESERVICE_API_ENDPOINT_URL=https://eservices-
test.vraa.lv/EservicePlatform.Examples.React.EP500Api/api
EP500_IMAGE_ESERVICE_URN=URN:IVIS:100001:EP-EXAMPLEDELEGATIONMVC-V1-0
#serviceIntegration parametri
SERVICEINTEGRATION_IMAGE_PUBLIC_URL=http://localhost:5006
SERVICEINTEGRATION_IMAGE_AUTH_CLIENT_ID=urn:oauth2:cs:complexui:react
SERVICEINTEGRATION_IMAGE_AUTH_CLIENT_SECRET=8C4E6AFC3B4D278521C2D8524A9B285C5B54C
28290C48D4D170F06931FF12998
SERVICEINTEGRATION_IMAGE_AUTH_SCOPE="openid profile epak context_api"
SERVICEINTEGRATION_IMAGE_ESERVICE_API_ENDPOINT_URL=http://localhost:5005
SERVICEINTEGRATION_IMAGE_ESERVICE_URN=URN:IVIS:100001:EP-
EXAMPLESERVICEINTEGRATIONMVC-V1-0
```

5. No Saknes direktorijas izpildīt komandu(servisa nosaukumu iespējams atrast docker-compose.yml datnē zem mainīgā services) - *docker-compose up --build <servisa nosaukums>*

4. Jauna e-pakalpojuma izveidošana

4.1. Atbalstītā LVP integrācija

Visos e-pakalpojumos uzsākot e-pakalpojuma izstrādi ir jāparedz šāda funkcionalitāte:

- **Mērogojams interfeiss**: e-pakalpojumiem jāatbalsta izmēra maiņu atbilstoši pārlūka loga platumam – visas ietvarā ietvertās komponentes nodrošina šādu funkcionalitāti. Veidojot e-pakalpojumu soļus e-pakalpojuma specifiskajiem HTML un CSS arī jānodrošina šo funkcionalitāti.
- **Vājredzīgo režīms**: e-pakalpojumiem jāatbalsta vājredzīgo režīmiem atbilstošu attēlojumu – visas ietvarā ietvertās komponentes nodrošina šādu funkcionalitāti. Veidojot e-paklpojumu soļus e-pakalpojuma specifiskajiem HTML un CSS arī jānodrošina šo funkcionalitāti.
- **Teksta palielināšana/samazināšana**: e-pakalpojumiem jāatbalsta iespēju palielināt un samazināt tā teksta izmēru.
- **Daudzvalodu atbalsts**: e-pakalpojumiem jābūt pieejamiem šādās valodās – latviešu, angļu krievu.

4.1.1. CSS canvas pārrakstīšana vājredzīgo režimos

Lai nodrošinātu CSS canvas pārrakstīšanu vājredzīgo režīmos un nodrošinātu bildes pārkrāsošanu, ir jaizveido jauns stila fails, vai jāievieto šis bloks esošaja failā. Zemāk redzamais piemērs attēlo ceļu, kā tiek nokrāsots elements ar klasi ".selector" vājredzīgo režimā, šis risinājums darbosies abām krāsam (yellow-on-black, un black-on-yellow). Svarīgi ir krāsam izmantot mainīgos "$primary" un "$secondary", kur ir nodefinētas pašas krāsas.

```
$colors: 'black-on-yellow', 'yellow-on-black';

$primary: '';
$secondary: '';

// without hashes
$primary-wh: '';
$secondary-wh: '';

@each $item in $colors {
@if $item == 'black-on-yellow' {
$primary: #f9f300;
$secondary: #000;
$primary-wh: 'f9f300';
$secondary-wh: '000';
}
@if $item == 'yellow-on-black' {
$primary: #000;
$secondary: #f9f300;
$primary-wh: '000';
```

```
$secondary-wh: 'f9f300';
}

.#{$item} {
.selector {
color: $secondary;
background: $primary;
}
}
}
```

4.2. Priekšnosacījumi e-pakalpojumu integrācijai

Lai e-pakalpojumu varētu veiksmīgi integrēt ar ārpus izstrādātāja datora esošo infrastruktūru (SSO, STS, utt.), nepieciešams veikt šādas darbības:

- Reģistrēt jauno pakalpojumu e-pakalpojumu reģistrā un iegūt e-pakalpojuma URN (iesniegums VRAA).
- Reģistrēt e-pakalpojuma *clientId* kā IDS uzticamo pusi (tiks nodrošināts ar e-pakalpojuma reģistrācijas iesniegumu VRAA, kurš minēts iepriekšējā punktā).

4.3. E-pakalpojuma projekta izveide un uzstādīšana

Šajā sadaļā aprakstīts kā izveidot un uzstādīt jaunu e-pakalpojuma piemēra projektu un tā komponentes, abus - gan SPA, gan MPA lietojumus.

4.3.1. SPA react projekta uzstādīšana

React vides parametru konfigurācija

React projekta darbībai nepieciešamie vides parametri tiek definēti `.env` vai `index.js` datnē. Piemēra parametri ir atrodami `.env.example` datnē. Mainīgos `index.js` var definēt `ConfigStore` projektā, kurā tiek izmantota `eservice-core` pakotne. Galvenā prioritāte ir `.env` datnē norādītajiem mainīgajiem, tad tiek ņemti vērā `ConfigStore` norādītie mainīgie, ja nav norādīti mainīgie abos failos tiek ņemta vērtība pēc noklusējuma.

Esošos parametrus var izdalīt pēc to funkcionālajām kategorijām:

- PUBLIC_URL - Norāde uz React saknes direktoriju (nepieciešams, ja risinājums tiek izvietots uz kāda no domēna apakšdirektorijam, piemēram, http://proxy.lv/subdirectory/). Šeit norāda tikai domēna apakšdirektoriju, nevis pilno adresi;
- environment - Vide (piemēram, development vai production);
- PUBLIC_PORT - Izmantotais ports;
- AUTH_[..] mainīgie atbild par autentifikācijas servisu, kur:
    - AUTH_AUTHORITY_API_ENDPOINT – Obligāts mainīgais, autentifikācijas API servisa galapunkts;
    - AUTH_CLIENT_ID - Obligāts mainīgais, autentifikācijas klienta identifikācijas numurs;
    - AUTH_CLIENT_SECRET - Obligāts mainīgais, autentifikācijas klienta secret kods;
    - AUTH_SCOPE - Obligāts mainīgais, autentifikācijas darbībai nepieciešamais datu tvērums;

- ESERVICE_[..] mainīgie atbild par e-pakalpojuma servisu, kur:
  - ESERVICE_API_ENDPOINT_URL - Obligāts mainīgais, E-pakalpojuma API servisa galapunkts;
  - ESERVICE_URN - Obligāts mainīgais, E-pakalpojuma unikālais identifikators;
- CONTEXT_API_ENDPOINT_URL - Obligāts mainīgais, konteksta API servisa galapunkts;
- NAVIGATION_API_ENDPOINT_URL – Navigācijas API servisa galapunkts;
- ASSETS_CDN_URL - Obligāts mainīgais, resursu bibliotēkas galapunkts;
- SEARCH_ENDPOINT_URL – Obligāts mainīgais, saite uz meklēšanas lapu
- EXECUTED_ESERVICE_URL – Izpildīto e-pakalpojumu adrese
- PORTAL_HOME_URL – Obligāts manīgais, saite uz portāla sākumlapu
- PORTAL_PROFILE_URL - Obligāts manīgais, saite uz portāla lietotāja profilu
- IDLE_LOGOUT_TIMEOUT_MINUTES - Laiks minūtēs pēc cik ilgas bezdarbības lietotāja tiks veikta pārbaude par aktivitāti

```
"[{"languageCode":"lv","text":"LV Title"},{"languageCode":"en","text":"EN
Title"}]"
```

Piemērs index.js norādītam mainīgajam:

```
// src/index.js

import {ConfigStore} from "@eserviceplatform/core";

ConfigStore.set({

      ESERVICE_TITLE: "My Custom Title"

});
```

5. E-pakalpojumu projekts

Šajā sadaļā aprakstīts E-pakalpojuma projekts SPA un MPA lietojumos.

Projektējot e-pakalpojumu jāpievērš īpaša uzmanība drošības aspektiem:

- SPA pakalpojumos kā papildus drošības slāni ieteicams izmantot BFF servisu.
- Veikt ievadīto datu validāciju ne tikai pārlūkā, bet arī BFF un biznesa servisā.
- Kur iespējams objektu identifikatorus un sensitīvus datus nepārsūtīt tīrā veidā uz pārlūku, piemēram, neizmantot personas kodus kā atslēgas sarakstos, tos ieteicams hešot, šifrēt aizstāt ar citu atribūtu, kas var unikāli identificēt objektu, vai kā citādi apstrādāt.
- No servera un /vai biznesa servisa uz pārlūku nodot tikai nepieciešamo informāciju.
- Biznesa servisos obligāti jāveic talona derīguma pārbaude.
- Biznesa servisos obligāti jāveic pieprasīto datu īpašnieka pārbaude - jāatgriež dati tikai ja tie pieder autentificētajam lietotājam.

5.1. Projekta struktūra

Šobrīd projektam paredzētas 2 veida struktūras, viena SPA lietojumam otra MPA. Projekta piemēriem šobrīd SPA lietojumam tiek izmantots .Net Core 3.1 un React tehnoloģijas, attiecīgi MPA variantā tā ir .Net Core 3.1.

Nepieciešamās zināšanas izstrādei abiem lietojumiem:

- .NetCore 3.1
- OIDC.
- Docker,
- Kubernetes
- Helm

SPA pieeja

Šajā variantā ir paredzēts, ka daļa no e-pakalpojuma biznesa loģikas atrodas klienta pusē un kā starpslānis saziņai ar biznesa servisiem kalpo BFF, kas realizēts kā atsevišķs serviss. Klienta pusē izveidotai lietotnei pašai ir jāspēj sazināties ar vismaz šādiem servisiem ContextApi, NavigationApi un Lvp.Portal.IdentityServer (IDS). Otra daļa no SPA pieejas ir starpslānis starp grafisko saskarni un infrastruktūru – BFF(backend for frontend), kas nodrošina sensitīvākas informācijas apstrādi un saziņu ar infrastruktūras un biznesa servisiem. Klienta pieprasījumi starp lapām notiek neveicot pilnas lapas pārlādi un saziņa ar serveri notiek izmantojot AJAX pieprasījumus.

Nepieciešamās zināšanas specifiski izstrādei SPA pieejā:

- React,
- Nginx

MPA pieeja

Atšķirībā no SPA pieejas, katrs klienta puses pieprasījums starp lapām notiek veicot pilnu pārlādi. Šajā pieejā ir paredzēta viena komponente kura pati arī nodrošina visu servisu izmantošanu. Tomēr servisi var tik izsaukti izmantojot AJAX tehnoloģijas.

Nepieciešamās zināšanas specifiski izstrādei MPA pieejā:

- Razor Pages

5.2. Lokalizācija

Kopīgās īpašības

Abos piemēros lokalizācijas valodas izvēle darbojās pēc šāda principa:

1. Tiek pārbaudīts vai saitē nav valodas <adrese>/**<valoda(lv, en, ru)>**/…..
2. Ja valodas nav saitē, tiek pārbaudīts vai valoda nav saglabāta sīkdatnēs
3. Ja valodas nav sīkdatnēs tiek izvēlēta latviešu valoda

Tulkojumus nodrošina divas yaml resursa datnes:

- Vispārīgie tulkojumi *global.yaml* – Satur vispārīgus tulkojumus kurus izmanto pats ietvars iebūvētajās funkcionalitātēs. Tiek uzturēts no VRAA puses.
- Epakalpojuma specifiskie tulkojumi *<epakalpojuma_nr>.yaml* – Satur izstrādātāja izveidotos tulkojumus, kuri nepieciešami konkrētā e-pakalpojuma funkcionalitātēm.

Tulkojumu failu struktūra ir yaml sekvence, skat piemēru zemāk:

```
- key: make_payment
  lv: "Veikt apmaksu"
  en: "Make payment"
  ru: "Оплатить"
- key: created
  lv: "Izveidots"
  en: "Created"
  ru: "Создано"
```

Augstāk minētās datnes attiecīgiem mērķiem un e-pakalpojuma stadijām ir pieejami šādi:

- Lokālai izstrādei gan Vispārīgo gan specifisko tulkojumu datnes pieejamas e-pakalpojuma piemēros, SPA pieejai – katra Starpslāņa piemēra direktorijā *Resources* un MPA pieejai katra piemēra direktorijā *Resources.*
- Tālākās vidēs Tulkojumi tiks izvietoti Attiecīgās vides Resursu glabātuvē un būs pieejamai sekojoši
    - Vispārīgie tulkojumi *<resursu_adrese>/Global/global.yaml*
    - Specifiskie tulkojumi *<resursu_adrese>/<epakalpojuma_nr>/<epakalpojuma_nr>.yaml*.

Ceļš uz resursu datnēm tiks apstrādāts pēc šādiem principiem

- Ceļš sākas ar https:// vai http://. Datne tiks lejupielādēta no norādītās adreses
- Cits ceļs. Sistēma mēģinās atvērt datni norādītajā ceļā

SPA pieejā ceļš uz tulkojuma datnēm jānorāda starpslāņa appsettings.json projekta datnē.

MPA pieejā ceļš uz tulkojuma datnēm jānorāda appsetings.json projekta datnē.

Abos variantos vides mainīgo parametri ir:

```
"TranslationGlobalUri": "Resources/global.yaml",
"TranslationUri": "Resources/serviceintegration.yaml",
```

Zemāk aprakstīts kā darboties ar tulkojumiem katrā pieejā.

SPA pieeja

Pieejamajos e-pakalpojumu SPA pieejas piemēros ir realizēti 2 veidi, kā tiek saņemti tulkojumi tekstiem.

1. Tulkojumi tiek iegūti no ContextApi un NavigationApi servisiem, šādi tiek ieviesti komponenti “Footer“ un “Header“. Piemēram, no NavigationApi header/footer servisiem nāk objekts {lv:{},en:{},ru:{}} un front-end puse attēlo datus attiecīgi no izvēletās valodas(skat. 7.10.1 nodaļu).
2. Tulkojumi tiek iegūti no e-pakalpojuma BFF komponentes. Piemērs - Lvp.EservicePlatform.Examples.React.ComplexUI, lapas ielādes sākumā sistēma pieprasa sarakstu ar tulkojumiem no servera:

BFF komponente atgriež visus unikālos tulkojumus, kuri definēti šajā e-pakalpojumā, React glabā tulkojumus *local storage*.

Tulkojumu iespējams norādīt pēc šāda šablona:

“TRANSLATIONS::{TRANSLATION_KEY}“

“TRANSLATIONS::“ - unikāls patterns, lai varētu atšķirt to, kas ir jātulko, un kas nav

“{KEY}“ - teksta atslēga

React piemēram ir izstrādāta komponente (i18n), kas parsē saņemtos tekstus, un atgriež tulkojumu. Visas metodes ar dokumentāciju var apskatīt **“e-pak piemera“** repozitorija **“/src/mixins/i18n.js“**

Pēc noklusējuma e-pakalpojuma paciņa automātiski meklē tulkojuma šablonu tekstos, bet gadījumā, ka teksts ienāk no cita resursa, to ir iespējams tulkot manuāli izsaucot “i18n.replaceKeysInString({key})“ metodi, tas atgriezīs iztulkotu tekstu. Lai izmantotu i18n funckcionalitāti ir nepieciešams iekļaut mixin “i18n.js“, no @eserviceplatform/fronted-react pakotnes.

Spa pieejā tulkojumus iespējams izgūt arī starpslāni. Lai to darītu nepieciešams inicializēt tulkojumu paplašinājumu. Lietojums zemāk piemērā.

```
var yamlFileGlobal = _config["Config:TranslationGlobalUri"];
var yamlExtGlobal = new YamlExtension(yamlFileGlobal);
var translationsGlobal = new TranslationCollection
{
      TranslationList = yamlExtGlobal.GetTranslations().ToList(),
}
var errText = translationsGlobal.GetTranslation("translation_key");
```

MPA pieeja

MPA piemēros tulkojumi skatos tiek ielādēti automātiski un tos iespējams izmantot izsaucot skata tulkojumu kolekciju:

```
<label>@Model.Translations.GetTranslation("translation_key")</label>
```

Tulkojumiem iespējams piekļūt arī skata modeļa *PopulateViewModels* metodē izmantojot Translations īpašību.

Tulkojumus var populēt arī manuāli izmantojot YamlExtension paplašinājumu, kā parādīts SPA pieejā.

5.3. Jauna soļa pievienošana

Lai pievienotu jaunu soli:

1. jāizpilda darbības, kas ir kopīgas, gan SPA gan MPA pakalpojumiem, skatīt 5.3.1;
2. jāveic darbības atkarībā no e-pakalpojuma tipa SPA skatīt 5.3.2, bet MPA skatīt 5.3.3;

5.3.1. Vispārējās darbības jauna soļa pievienošanai

Šajā sadaļā aprakstītās taš darbības, kuras jāveic neatkarīgi no tā vai tiek izmantota SPA vai MPA pieeja

- Soļa izveide

Katra projektā(spa pieejā bff projekta) ir datne */Models/Step.cs*, kurā ir klase *Step* kas implementē *IStep* saskarni un kalpo kā bāzes klase e-pakalpojuma soļiem.

Lai izveidotu jaunu soli var kopēt kādu soļa klases failiem iekš attiecīgā piemēra(spa variant domāts bff) *Models/Steps/* direktorijas un attiecīgi pārsaukt vērtībās. Kā piemēros redzams, katrs solis manto iepriekšminēto klasi *Step*. Zemāk soļa klases īpašību apraksts.

Īpašības nosaukums	Tips	Apraksts
urn	Simbolu virkne	Soļa identifikators
title	TextObject masīvs	Satur masīvu ar teksta objektiem, katrs masīva ieraksts atbilst vienai valodai(lv, ru, en). Vērtībā iespējams norādīt šādus vietturus: • MPA - <translation_key> : izmantos soļa nosaukumu no tulkojumiem • SPA - TRANSLATIONS::<translation_key> : izmantos soļa nosaukumu no tulkojumiem
nextStepUrns	Simbolu virkne	Nākamo soļu urn atdalīti ar ‘,’
nextSteps	Saraksts ar soļiem	Saraksts ar nākamajiem soļiem
validationRules	Lauka noteikumu saraksta vārdnīca	Visi validācijas noteikumi šim solim, lai solis tiktu pozitīvi validēts, tad visiem šajā īpašībā definētajiem noteikumiem ir jāizpildās, pretējā gadījumā e-pakalpojums atgriezīs kļūdu ar paziņojumu. Vārdnīcā atslēga ir simbolu virkne un tajā jānorāda lauka nosaukums (name) vai arī “general”, ja tā ir vispārīga validācijas kļūda. Attiecīgi kļūdas gadījumā kļūdas paziņojums tiks atgriezts zem norādītās atslēgas. Lauka noteikumi ir objekts, kurš satur vārdnīcu(rules) ar predikātu kopu šim laukam, piemēram, lauks ‘email’ var saturēt noteikumus -> ‘required’,’is_email’. Šobrīd sistēmā nav

		piedefinētu predikātu un tie ir jādefinē izstrādātājam. Šī objektā otra īpašība ir vārdnīca ar kļūdu paziņojumiem(messages). Lai atrastu attiecīgo kļūdu paziņojumu, tā atslēgai jāsakrīt ar attiecīgo atslēgu predikātu vārdnīcā. Visi predikāti ieejā saņem sarakstu ar no soļa iesūtītajiem laukiem, kas ir FieldValue objekts.
entryConditions	Predikāts	Šī īpašība nepieciešama, definētu, ka lietotājam ir iespējams pāriet uz šo solu. Pēc soļa validācijas un biznesa procesu izpildes, e-pakalpojums aprēķina nākamo soli. Ja solim nav nākamā soļa un tas nav pēdējais solis, tad tiks atgriezta kļūda. E-pakalpojums iterē cauri visiem iespējamajiem šī soļa nākamajiem soļiem un tiklīdz kā kāda soļa predikāts ir veiksmīgs vai predikāts nav norādīts tā pārvada lietotāju uz šo soli. Ja pēc predikātu pārbaudes neviens solis neiztur pārbaudi, tad tiek atgriezta sistēmas kļūda. Predikāts ieejā saņem e-pakalpojuma laikā aizpildītos laukus kā FieldValue sarakstu.
processStep	Funkcija	Tiek izmantots biznesa procesu veikšanai pēc soļa formas saņemšanas. Izpildās uzreiz pēc soļa validācijas Īpašībā saņem visas līdz šim e-pakalpojumā iesūtītās lauku vērtībās. Funkciju var izmantot asinhronām darbībām un validācijas kļūdu attēlošanai. Atgriež *Dictionary<string, string[]>*
components	Komponenšu masīvs	Paredzēts react komponenšu pievienošanai

- Soļa ievietošana soļu kolekcijā

Lai e-pakalpojums atpazītu un skatītu soli to ir nepieciešams pievienot soļu kolekcijai. Soļu kolekcija ir atrodama attiecīgā projekta(spa pieejā ir domāts bff projekts) datnē *Models/Steps/ViewModels/<prefix>StepCollection.cs*

Šajā datnē jābūt klasei, kura implementē *IStepCollection* saskarni. Attiecīgi klasē ir metodes *GetSteps* un *GetLinkedSteps*, kuras abas atgriež IStep masīvus, tikai pirmajā gadījumā tas ir pats solis, otrā soļiem ir aizpildīts nexSteps līdz pat e-pakalpojuma beigām. Efektīvākais veids kā aizpildīt šos masīvus ir izveidotajās soļu izveidot konstruktoru, kurš atkarībā no padotajiem parametriem atgriež tikai šo soli vai visus saistītos soļus. Konstruktora definīcijas Piemērs(bool linked nosaka vai atgriezt pilnu sarakstu vai tikai pašu soli):

public Step2a(IConfiguration config, IHeaderDictionary headers, bool linked = false)

5.3.2. SPA pieeja

Ir izstrādāts soļu konstruktors, kas ļauj būvēt jebkuru saturu e-pakalpojumu soļiem, faktiski soļu saturu ir iespējams pilnībā konfigurēt servera pusē, React tikai inicializē komponentes un pievieno papildus parametrus un funkcijas, lai komponentes darbotos pareizi.

Javascript pakotnē frontend-react ir izstrādāta funkcionalitāte, kura apstrādā no starpslāņa padotās komponentes “method” lauku. Šajā laukā var norādīt react pusē programmētu palīgkomponenti, kura piešķirs papildus funkcionalitāti komponentei, kura ir padota no starpslāņa. Šīs palīgmetodes var izmantot arī tam lai formu pilnībā programmētu react pusē.

Javascript pakotnē frontend-react ir izstrādātas palīgkomponentes visām “ReactSDK”(skat dokumenta 6. Punktu Bibliotēkas e-pakalpojumu izstrādei) komponentēm, kuras var izmantot un tas ir demonstrēts “ComplexUI” piemērā.

Papildus komponentēm no bibliotēkas šis skripts var parādīt parastus HTML elementus, jūs varat konstruēt un modificēt komponentu struktūru kā vēlaties, varat tos sasaistīt - izmantojot React Refs un parentContext (aprakstu skatiet zemāk).

Objekta “Component” parametru saraksts servera pusē. Component klase pieejama EserviceCore nuget pakotnē, vairāk skatīt šī dokumenta 6.punktā - Bibliotēkas e-pakalpojumu izstrādei):

Nosaukums (atslēga)	Vērtība	Obligāts	Apraksts
name	String	jā	Komponentes nosaukums no “ReactSDK“, vai jebkura html birka (komponenta inicializācijas prioritātes ir norādītas zemāk)
props	Object (dictionary)	nē	React props, objekts ar komponentu parametriem, ņemiet vērā, ka visām vērtībām ir simbolu virknes tips un, lai komponents tiktu pareizi attēlots (tiktu aizpildītas tādas propu vērtības kā metodes), tie ir jāpārvērš pareizajā tipa “helpera“ iekša (piemērs ir paradīts zemāk).
children	Array of Components (Component[])	nē	React Children, komponentu masīvs
html	String	nē	html simbolu virkne – iespējams izmantot lauka - children vietā. Ja īpašība children netiks norādīts, tad tiks izmantota īpašība html
method	String	nē	helpera nosaukums Reacta, kurš apstrādās šo komponentu

Zemāk ir ButtonGroup komponentes piemērs, kur iekša ir divas Button komponentes

```
new Component{
    name="ButtonGroup",
    children= new Component[]{
        new Component{
            name="Button",
            props=new Dictionary<string, string>{
                {"style","default"},
                {"bold","true"},
                {"rounded","true"},
                {"type","button"},
                {"wide","true"},
            },
            method="prevStepBtn",
            children= new Component[]{
                new Component{
                    html="Atpakaļ"
                }
            }
        },
        new Component{
            name="Button",
            props=new Dictionary<string, string>{
                {"style","danger"},
                {"bold","true"},
                {"rounded","true"},
                {"type","submit"},
                {"wide","true"},
            },
            children= new Component[]{
                new Component{
                    html="TRANSLATIONS::next_step"
                }
            },
        },
    },
},
```

Augstāk redzamajam komponentes piemēram, komponentei “Buttongroup” ir nodefinēts tikai “name” un “children”, kur laukā ‘name’ ir norādīts komponentes nosaukumu no ReactSDK.

Iepriekšminētā komponente “ButtonGroup” sevī iekļauj divas citas (abas “Button”) komponentes, tās ir vienādas, izņemot vienu parametru, pirmajam komponentam tiek piešķirts “method” parametrs ar vērtību “prevStepBtn”, tas nozīmē, ka React starp pieejamajām palīgkomponentēm meklēs metodi “prevStepBtn”:

Palīgkomponentes veidojās no frontend-react pakotnē definētajām noklusētajām un E-pakalpojuma izstrādātāja izstrādātajām palīgkomponentēm. Izstrādātāja palīgkomponenšu sarakstu nepieciešams iestatīt pirms frontend-react pakotnes inicializācijas(papildus detaļas skatīt e-pakalpojuma piemēros)

```
const helperNamings = {
    'ButtonGroup': HandleButtonGroup,
    'prevStepBtn': HandlePrevStepBtn,
};
```

```
ConfigStore.set({
    // papildu palīgkomponenti
    AdditionalHelpers: helperNamings,
});
```

Komponentā kurā norādītā metode “prevStepBtn”, tās attiecīgā palīgkomponente būs “HandlePrevStepBtn”(nosaukumu vārdnīcu jāveido react projekta datnē, kura atrodama src/helpers/index.js), kas pats par sevi ir cita komponente, kura izmanto “Button” komponenti ietvars, kurš apstrādā padotās vērtības no servera puses un pievieno papildus funkcionalitāti react pusē:

```
const {Button} = ReactSDK.Components;
import React, {Component} from "react";

export default class handlePrevStepBtn extends Component {
    constructor(props) {
        super(props);
    }

    render() {
        const {parentContext, rounded, bold, wide, ...rest} = this.props.options || {};

        const correctProps = {
            rounded: (rounded === 'true'),
            bold: (bold === 'true'),
            wide: (wide === 'true'),
            ...rest
        };

        return (
            <Button {...correctProps} onClick={()=>{
                parentContext.navigateStep( index: '-1');
            }}>
                {this.props.children}
            </Button>
        );
    }
}
```

Otra poga, kura minēta piemērā nesatur parametru “method” , līdz ar ko tā darbojās saskaņā ar ReactSDK definīciju un saņemtajiem īpašību parametriem. Šajā gadījumā tā iesniedz pogu

Katrai palīgkomponentei ir šādi parametri:

Prop name	Type	Description
parentContext	Component instance	Komponenta "Step.js" instance, lai jūs varētu izsaukt metodi no e-pakalpojuma galvenās palīgkomponentes konteksta
options	Object	Parametri, kas nāk no servera puses(vērtības vienmēr būs simbolu virknes, kuras jāpārvērš pareizajos tipos, skatiet tālāk, kā tās var pārveidot)
children	Object|Array	react children

Tā kā visas “options“ objekta vērtības ir simbolu virknes, tās ir jāpārvērš pareizajos tipos. Piemēram, pogai ir “rounded” parametrs, kuras tips ir “Boolean”, šis parametrs jāpārveido no simbolu virknes uz Boolean vērtību, jūs varat izstrādāt un izmantot savu konvertēšanas metodi, bez Boolean vērtībām. Piemēros, tas ir realizēts šādi:

```
const {parentContext, rounded, ...rest} = this.props.options || {};

const correctProps = {
    rounded: (rounded === 'true'),
    ...rest
};

return (
    <Button {...correctProps} onClick={()=>{
```

“(rounded === “true”)“ atgriezīs “true” vai “false” jau ar Boolean tipu

Komponenta nosaukuma prioritātes (nosaukums):

1. Galvenā prioritāte ir objekts "HelperNamings", kas atrodas failā "initComponent.js". Šajā objektā ir definētas visas iespējamās palīgkomponentes šajā projektā, kuras izmanto, lai apstrādātu atgriezto komponenšu kopu.
2. ja "HelperNamings" nesatur vērtību no "method" vai "name", tad palīgkomponente tiek ņemta no "ReactSDK" komponentu saraksta, tas pats princips attiecas arī uz children komponentiem
3. ja "name" vērtība nav komponenta nosaukums no "ReactSDK", tad šī vērtībā tiks uzskatīta kā html birka

Piezīmes:

Lokalizācija:

Komponentu inicializēšana ir arī cieši saistīta ar lokalizāciju, parametrus kur ir norādīti daži teksti, var definēt ar lokalizācijas patterniem (skat. punktu 5.2)

Piemērs:

```
new Component{
    name="Button",
    props=new Dictionary<string, string>{
        {"style","danger"},
        {"bold","true"},
        {"rounded","true"},
        {"type","submit"},
        {"wide","true"},
    },
    children= new Component[]{
        new Component{
            html="TRANSLATIONS::next_step"
        }
    },
},
```

komponentes "Button" saturs ir parasta HTML simbolu virkne, kuras vērtība ir unikāla tulkošanas atslēga, kas ir norādīta tulkojumu resursu failos, kuri piemēros ir atrodas, katra projekta direktorijas "Resources" direktorijā

```
public Translation[] GetTranslations()
{
    return new Translation[] {
        new Translation{Key="next_step",Lv="Tālāk",En="Next",Ru="Дальше"}
    };
}
```

Pārējo darbu frontends veiks pats, tas šim parametram piešķirs jaunu vērtību un arī valodas atjaunināšanas gadījumā to atjaunos (SPA).

Specifiska komponenta helpera vietā jūs varat arī izveidot palīgu, kurā būs pilnvērtīgs "solis", kas sastāv no daudziem komponentiem.

Servera puse:

```
children= new Component[]{
    new Component {
        html="Šī soļa saturs tiek definēts tikai frontend puse, no backenda nāk tikai šis teksts un iesaņojuma bloks."
    },
    new Component {
        name="div",
        method="stepDataOnFrontend"
    }
}
```

stepDataOnFrontend palīgkomponente ("render" metode):

```
render() {
    return (
        <Form onSubmit={this.formOnSubmit}>
            <div className="w-100">
                <div className={'mt-3'}>
                    <h3>Komponenšu saraksts un kontents, atrodas tikai frontend puse!</h3>
                    <p className={'mt-1'}>Visa struktūra un komponenti tiek definēti frontenda</p>
                </div>
                <ul>
                    <li className={'mt-3'}>
                        <div className={'color-gray-300 mt-4 mb-2'}>Ievadlauks</div>
                        <div>
                            <InputForm name={'input_fs_1'} placeholder={'kaut kāds placeholders'} />
                        </div>
                    </li>
                    <li className={'mt-3'}>
                        <div className={'color-gray-300 mt-4 mb-2'}>Lokalizacijas piemers (nomainiet valodu)</div>
                        <div>
                            <span>{i18n.getTranslation( key: 'primary_text')}</span>
                        </div>
                    </li>
                    <li className={'mt-3'}>
                        <div className={'color-gray-300 mt-4 mb-2'}>Pazinojums</div>
                        <div>
                            <Alert
                                hide={this.state.alertState}
                                onHide={
                                    () => {
                                        this.setState( state: {
                                            alertState: true
                                        })
                                    }
                                }
                                onShow={
                                    () => {
                                        this.setState( state: {
                                            alertState: false
                                        })
                                    }
```

Šajā gadījumā no servera nāk tikai viena komponente, kas kopumā veido visu e-pakalpojuma soli.

Šādas sistēmas priekšrocība ir tā, ka jūs varat izveidot unikālas palīgkomponentes un apvienot tās kopā, nevis pastāvīgi kopēt. Pietiek ar to, lai šie apstrādātāji tiktu ieviesti vienreiz - un tos izmantotu jebkurā e-pakalpojuma posmā.

Soļus nepieciešams pievienot arī starpslānī, to izveidi skatīt punktā 5.3.3 sadaļā Soļa skata modeļa izveide

5.3.3. MPA pieeja

- Soļa skata modeļa izveide

MPA variantā datu atveidošanai tiek izmantoti razor šabloni, kuros ir ērti strādāt, ja tiem ir definēts modelis. Līdz ar ko MPA variantā tajā pašā datnē kurā ir izveidota attiecīgā soļa klase, jāizveido arī soļa moduļa klase, kurā var norādīt visus nepieciešamos mainīgos priekš šī skata. Zemāk piemērs no MPA soļa faila, tas atrodams arī ComplexUI piemērā.

```
public class Step6a : Lvp.EservicePlatform.Examples.Mvc.ComplexUI.Models.Step
{
      public Step6a(IConfiguration config, IHeaderDictionary headers, bool linked
= false)
      {
            this.urn = "step6a";
```

```
            this.title = new TextObject[] {
                        new TextObject
                        {
                              LanguageCode ="lv",
                              Text = "first_step_title"
                        },
                        new TextObject
                        {
                              LanguageCode ="en",
                              Text = "Sixth Step"
                        },
                        new TextObject
                        {
                              LanguageCode ="ru",
                              Text = " шестой шаг"
                        }
                  };
            this.nextStepUrns = "complete";
            this.validationRules = null;
            this.entryConditions = null;
            this.processStep = progress =>
            {
                  List<FieldValue> x = progress;
                  LvpContext services = new LvpContext(config, headers);
                  return true;
            };
            if (linked)
            {
                  this.nextSteps = new List<IStep>
                  {
                        new Complete(config, headers, linked),
                  };
            }
      }

      // nepipeciešams aizppildīt tikai SPA pieeja
      public static Component[] GetComponents(IConfiguration config,
IHeaderDictionary headers)
      {
            return new Component[] { };
      }
}

public class Step6aViewModel
```

```
{
      public Step6aViewModel(ComplexUIStepModel
stepModel,List<EServiceWizardStep> progress)
      {
            this.stepNavigationButtons = new ButtonGroup();
            this.stepNavigationButtons.AddItem(
                  new Button(stepModel.Translations.GetTranslation("prev_step"))
                        .Style(ButtonStyle.Default)
                        .DataAttr("name='back' value='true'")
                        .Rounded()
                        .Bold()
                        .Size(ButtonSize.Medium)
                        .Wide()
            );
            this.stepNavigationButtons.AddItem(
                  new Button(stepModel.Translations.GetTranslation("next_step"))
                        .Style(ButtonStyle.Danger)
                        .Rounded()
                        .Bold()
                        .Size(ButtonSize.Medium)
                        .Wide()
                        .Type(ButtonType.Submit)
            );
      }
      public ButtonGroup stepNavigationButtons { get; set; }
}
```

- Soļa skata modeļa populēšana

Pēc tam kad ir izveidots soļa modelis. To nepieciešams populēt ar datiem un pievienot galvenajam soļu modelim(datne atrodama katra projekta */Models/ViewModels* direktorijā), kurā ir pieejamas arī dažādas īpašības no paša e-pakalpojuma. Šajā module var pievienot arī izveidotos soļu modeļus. Attiecīgi E-pakalpojuma nuget pakotne izsauc šī modeļa interfeisa metodi PopulateViewModels ar parametriem, kuri satur sistēmas konfigurāciju, pieprasījuma galveni un esošā e-pakalpojuma progresu ar aizpildītajiem laukiem un tajā ir paredzēts populēt iepriekš pievienotos soļu modeļus. Atkal – ir ērti soļu skata modeļiem veidot konstruktorus kuri pieņem nepieciešamās vērtības un aizpilda soļa modeļa īpašības. Zemāk aprakstītas skatā pieejamās īpašības:

īpašības nosaukums	tips	Apraksts
Language	string	Pašlaik aktīvā valoda (lv, en, ru)
Auth	bool	Pazīme vai lietotājs ir autorizēts
Size	int	Pieejamības teksta izmērs
Theme	string	Pieejamības krāsu tēmu
AssetsUri	string	Bāzes ceļš uz bildēm assetos
TabId	string	Esošais TabId
LvpContext	LvpContext	LvpContext servisi
Config	IConfiguration	E-pakalpojuma konfigurācija no servisa

EServiceConfig	Config	Sistēmas konfigurācija
NavbarItems	List<INavbarItem>	E-pakalpojuma navigācija
FooterTopColumns	List<IFooterColumn>	E-pakalpojuma kājenes augšējās kolonnas
FooterBottomColumns	List<IFooterColumn>	E-pakalpojuma kajenes apakšējās kolonnas
Breadcrumbs	List<IBreadcrumbItem>	E-pakalpojuma ceļs
Instructions	List<IAccordionItem>	E-pakalpojuma instrukcijas soļi
VideoInstruction	IVideo	E-pakalpojuma video instrukcijas palīgkomponente
Languages	string[]	Visas pieejamās valodas
SearchControl	SearchControl	E-pakalpojuma meklēšanas kontrolis
AlertControls	List<Alert>	E-pakalpojuma paziņojuma kontrolis
Suggestions	List<List<string>>	E-pakalpojuma meklēšanas ieteikumu
LoginControl	LoginControl	E-pakalpojuma autentificēšanās kontrolis
AuthUser	AuthUser	Autorizētais lietotājs
UserCardUsers	List<IUserCardUser>	E-pakalpojuma lietotāju kartiņas
UserCardLinks	List<IUserCardLink>	E-pakalpojuma lietotāja darbība
StepUrn	string	E-pakalpojuma aktīvā soļa urn
EServiceWizard	EServiceWizard	E-pakalpojuma objekts
Steps	string	E-pakalpojuma soļu saraksts
ValidationErrors	Dictionary<string, string[]>	Kļūdas paziņojumu vārdnīca
BaseUrl	string	Sistēmas bāzes url
Translations	TranslationCollection	Tulkojumu kolekcija
Helpers	Helpers	Palīgmetodes
PortalHomeAddress	string	LVP adrese
PaymentId	String	Maksājuma id, tiks padots uz skatu, pēc maksājuma apstiprinājuma

- Soļa skata izveide

Lai izveidotu skatu priekš modeļa var kopēt kādu no piemēros esošajiem skatiem, attiecīgā projekta *Views/Steps* direktorijā . Skata nosaukums ir jāveido pēc šāda principa <soļa_urn>.cshtml. Skatā vēlams izmantot e-pakalpojumā nodefinēto kopējo skata modeli. E-pakalpojuma solim ir jāizmanto "_Service" šablons, kurš pieejams nuget pakotnē.

5.4. Eservice-core(SPA) pakotnes palīgfunkcijas un komponentes

Eservice-core pakotnē ir pieejamas palīgfunkcijas un noklusējuma komponentes, kuras React projektā var izsaukt, importējot nepieciešamo funkcionalitāti izmantotajā komponentē. Noklusējuma komponentes tiek padotas caur objektu DefaultHelperList, kamēr palīgfunkcijas izsauc, no pakotnes izsaucot objektu ar nosaukumu mixins.

Importējot eservice-core pakotni, tiek izgūta šāda objektu struktūra:

```
{
    DefaultHelperList,
    AppWrapper,
    MainWrapper,
```

```
    ConfigStore,
    mixins,
    contexts,
    services,
    HelperComponent
}
```

- DefaultHelperList - Sevī ietver visas pakotnes noklusējuma komponentes, kas ir norādītas pakotnē;
- AppWrapper - Lietotnes apvalks; atbild par lietotnes maršrutētāju uzbūvi;
- MainWrapper - Galvenais apvalks; atbild par vides mainīgo pārbaudi un projekta uzbūvēšanu;
- ConfigStore - Konfigurācijas datu izgūšanas un uzstādīšanas funkcionalitāte; izmanto, lai izgūtu konfigurācijas parametrus no pakotnes un definētu jaunus atbilstoši vajadzībām;
- mixins - Sevī ietver visas palīgfunkcijas, kas ir nepieciešamas e-pakalpojuma darbībai, piemēram, pieprasījumi, SessionStorage funkcijas, sīkdatnes funkcionalitātes, u.t.t.;
- contexts - Ietver visus projekta darbībai nepieciešamos React kontekstus (par React kontekstiem vairāk lasīt šeit: https://reactjs.org/docs/context.html)
- services - Ietver visus servisus, kas saistīti ar pieprasījumu izsaukšanu uz ārējiem resursiem;
- HelperComponent - palīgkomponente, kas sevī ietver soļa kontekstu; izmanto, ja ir nepieciešams mantot vecāka (t.i., soļa) ;

ConfigStore tiek izmantots e-pakalpojuma konfigurāciju uzstādīšanai. Caur ConfigStore var uzstādīt:

- AdditionalHelpers: E-pakalpojumam nepieciešamās papildus React komponentes, kas nav pieejamas eservice-core pakotnē. Šeit padod JSON formāta mainīgo, kur atslēga ir komponentes nosaukums, bet vērtība - pati React komponente. Lai ērtāk eksportētu visas papildus komponentes, ir ieteicams visas komponentes definēt un eksportēt caur index.js failu, kas atrodas React komponenšu mapītē. Piemērs redzams zemāk:

```
// src/helpers/index.js
// Visas komponentes atrodas kā .js faili iekš /helpers/ mapītes

import HandleAutoCompleteDropdownForm from "./HandleAutoCompleteDropdownForm";
import HandleButtonDynamicChange from "./HandleButtonDynamicChange";
import HandleDataGridWithAjaxAndPagination from
"./HandleDataGridWithAjaxAndPagination";
import HandleFileUploadDeletable from "./HandleFileUploadDeletable";

export const AdditionalHelpers = {
      'HandleAutoCompleteDropdownForm': HandleAutoCompleteDropdownForm,
      'HandleButtonDynamicChange': HandleButtonDynamicChange,
      'HandleDataGridWithAjaxAndPagination': HandleDataGridWithAjaxAndPagination,
      'HandleFileUploadDeletable': HandleFileUploadDeletable,
}

// Komponenšu definēšana iekš ConfigStore
```

```
// src/index.js

import { AdditionalHelpers } from "./helpers";
import {ConfigStore} from "@eserviceplatform/core";

ConfigStore.set({
      AdditionalHelpers: AdditionalHelpers,
})
```

- AdditionalPages: E-pakalpojumam nepieciešamās papildus lapas ar sev atbilstošo maršrutu. Var izmantot, ja ir nepieciešams definēt papildus lapas e-pakalpojumam. Piemērs:

```
// src/index.js
import {ConfigStore} from "@eserviceplatform/core";

ConfigStore.set({
      // define additional pages
      // key - page url (path)
      // value - Route props (see the react-router-dom docs)
      AdditionalPages: {
            'custompage': {
                  exact: true,
                  render: (props) => {
                        return (
                              <div>
                                    Additional page body
                              </div>
                        )
                  }
            }
      },
})
```

- yupValidationSchemas: yup validācijas shēmas. Eservice-core komponente HandleFormWithFormik izmanto Formik apvienojumā ar yup bibliotēku, lai nodrošinātu formu validāciju. Ja ir nepieciešams definēt šai formai specifiskas validācijas, tās var padot caur ConfigStore. Vairāk par yup var lasīt šeit: https://github.com/jquense/yup. yup validācijas shēmas definēšanas piemērs iekš ConfigStore:

```
// src/index.js
import {ConfigStore} from "@eserviceplatform/core";

ConfigStore.set({
      yupValidationSchemas: {
            'validationSchemaFirst': Yup.object().shape({
```

```
                    'input_0':      Yup.string().required('it      is      required
                    field').min(3,'input    should    contains    at    least    ${min}
                    chars').max(10,'exceeded input chars limit ${max}'),
                    'input_1': Yup.string().matches(regexPatterns['equalsLatvija'],
                    'value must be equals "Latvija.lv"'),
                    "date_0":     Yup.string().matches(regexPatterns['equalsDate'],
                    'value must be equals "26-07-2020"'),
                    'combobox_0':
                    Yup.string().required().matches(regexPatterns['equalsSomething'
                    ], 'value must be equals "something"'),
                    'autocomplete_0':
                    Yup.string().required().matches(regexPatterns['equalsSomething'
                    ], 'value must be equals "something"'),
                    "multiselect_0": Yup.mixed().test({
                          message: 'this input value should contains "["1","2"]"',
                          test: (value) => {
                                if(!value) return false
                                return  value.includes("1")  &&  value.includes("2")
                                && value.length === 2
                          }
                    })
              }),
              'validationSchemaSingleField': Yup.object().shape({
                    'input_2':  Yup.string().matches(regexPatterns['personalCode'],
                    'Wrong personal code format'),
              })
        }
})
```

Objekts mixins sevī ietver vairākas palīgfunkcijas, kuras var funkcionāli izdalīt:

- Cookies - Ietver sevī funkcijas sīkdatņu izgūšanai, uzstādīšanai, dzēšanai;
- FileStorage - Ietver sevī funkcijas datņu apstrādei un izgūšanai;
- i18n - Atbild par lietotnes lokalizāciju;
- SessionStorage - Ietver funkcijas sesiju izgūšanai, dzēšanai, u.c.;
- requests - Ietver dažādu pieprasījumu veidu (POST, GET) izsaukšanas funkcijas.
- getCorrectTypedProps - Izmanto, lai formatētu no backend-a saņemtās prop vērtības uz pareiziem tipiem;
- getCorrectTypedValue - Izmanto, lai formatētu no backend-a saņemtās vērtības uz pareiziem tipiem;
- StringFormatter – Izmanto, lai formatētu virknes;

Piemēri mixins funkciju izsaukšanai React komponentē:

```
import {mixins} from "@eserviceplatform/core"


mixins.requests.getRequestHeaders(x);
mixins.SessionStorage.set(x,y);
```

```
mixins.LocalStorage.getLocalSavedData();
mixins.FileStorage.removeFile(y);
```

Objekts contexts sevī ietver sekojošos kontekstus:

- ParentContext: Komponentes Step konteksts, kuru pēc nepieciešamības izmanto e-pakalpojumā izmantotās komponentes.

Ja React projektā komponentes definē kā klašu komponentes (class components), tad soļa konteksta izgūšanai izmanto HelperComponent komponentes mantošanu

```
const {
      HelperComponent
} = require("@eserviceplatform/core")

export default class CustomComponent extends HelperComponent
```

Ja React projektā komponentes definē kā funkcionālās komponentes (functional components), tad soļa konteksta izgūšanai izmanto ParentContext palīgkontekstu.

```
const {
      contexts: {
            ParentContext
      }
} = require('@eserviceplatform/core')

const HandleDataGridWithAjaxAndPagination = (props) => {
      const stepContext = useContext(ParentContext);
      const someFunction() {
            stepContext.toggleLoader(false);
      }
}
```

Objects services sevī ietver servisu sekojošos izsaukumu objektus:

- SessionService - Ietver visus pieprasījumus uz sesijas API servisu;
- DocumentService - Ietver visus pieprasījumus uz dokumentu API servisu;
- ProfileService - Ietver visus pieprasījumus uz profila API servisu;

Servisu izmantošanai React komponentēs ir nepieciešams tās inicializēt kā jaunu instanci, piemēram:

```
this.sessionService = new SessionService
this.sessionService.getSessionAttributes()
```

Lai datnē funkciju izsaukšanai neizmantotu objektu prefiksus, importēšanas brīdī tās ir nepieciešams destrukturizēt:

```
const {
      mixins: {
            requests: {getData},
            SessionStorage
```

```
	},
	DefaultHelperList: {
		InputForm
	},
		contexts: {
			ParentContext
	},
	services: {
		SessionService
	}
} = require("@eserviceplatform/core")

getData(x,y);
SessionStorage.get('x-tabId');
<InputForm />
static contextType = ParentContext;

this.sessionService = new SessionService
this.sessionService.getSessionAttributes()
```

5.5. Publiski pieejamās komponentes (SPA)

Step.js – komponente, kas būvē e-pakalpojuma soļu lapu. Piekļūst caur “ParentContext” kontekstu.

Publiskās metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
setAlertDefaultMsgs(arr)	arr - String[]	-	Iestatīt alert paziņojumus
async getFileStorage(name, payload)	name - String, payload - Array	Promise<*>	Iegūst faila krātuves instances
async renderErrorAsAlert(errors)	errors - String[]	Promise<void>	Attēlot kļūdas kā alert paziņojumus
setFieldValidationErrors(errors)	errors - Object	Promise<void>	Iestatīt validācijas kļūdas laukiem
returnScrollToTop()	-	-	Ritina logu [atpakaļ] uz augšu pa y asi
async navigateStep(index, form)	index - String, form - HTMLElement	Promise<void>	Pārslēgties uz nākamo/iepriekšējo soli

submitStep(data)	data - Object	-	Iesniedz soli un pāriet uz nākamo
goToPrevStep()	-	-	Pāriet uz iepriekšējo soli
toggleLoader(bool)	bool - Boolean	-	Kontrolē ielādēšanas komponenti
async toggleLoaderAsync(bool)	bool - Boolean	Promise<void>	Asinhroni kontrolē ielādēšanas komponenti
getCurrentStepFieldValues()	-	Array	Atgriež pašreizējā soļa saglabāto lauku vērtības
getFieldValueByKey(key)	key - String	Object	Atgriež lauka vērtību pēc atslēgas
async switchStep(urn)	urn - String	Promise<void>	Pāriet uz soli ar norādīto urn vērtību

5.6. Servisi (MPA)

MPA lietojumam pieejama klase LvpContext, kura satur 7. punktā aprakstīto servisu pieprasījumu metodes, kuras iespējams izsaukt no E-pakalpojuma.

Servisu konstruktors :

```
LvpContext services = new LvpContext(config, headers);
await services.Request.StopTransaction(transactionid);
```

Konstruktora parametri:

Tips	Nosaukums	Apkrraksts
IConfiguration	config	E-pakalpojuma vides parametri (Appsettings)
IHeaderDictionary	headers	Pieprasījuma galvenes parametri
boolean	authrequired	Pazīme par to vai šī servisu instance nodrošina autentificētu servisu lietojumu. Noklusētā vērtība - false

Servisu izsaukums :

```
LvpContext services = new LvpContext(config, headers);
await services.Request.StopTransaction(transactionid);
```

Pieejamo servisu apraksts:

SessionProperties
GetAllSessionAttributesAsync():Task<Dictionary<string,string>>
GetSessionAttributeAsync(string key):Task<string>
SetSessionAttributeAsync(string key, string value):Task
DeleteSessionAttributeAsync(string key):Task
Request
StartTransaction():Task<string>
StopTransaction(string transacionId):Task<bool>
IvisRequest(StringContent content, string targetUrl):Task<JObject>
ApiRequest(StringContent content, string targetUrl, ApiRequestMethod httpMethod = ApiRequestMethod.Post):Task<HttpResponseMessage>
EdkService
CreateDocumentAsync(IEnumerable<Property> properties, Dictionary<string,string[]> additionalAces, StreamPart streampart):Task<string>
GetDocumentAsync(string documentId, string filter):Task<IEnumerable<Property>>
GetDocumentContentAsync(string documentId):Task<HttpContent>
GetDocumentsAsync(string filter, uint? skip, ushort? take):Task<List<IEnumerable<Property>>>
ShareDocumentInFolderAsync(string documentId, string path):Task
StopSharingDocumentInFolderAsync(string documentId, string path):Task
UpdateDocumentAsync(string documentId, IEnumerable<Property> properties):Task
UpdateDocumentContentAsync(string documentId, StreamPart streampart):Task
UserProfileService
GetPropertiesAsync():Task<IEnumerable<TypedProperty>>
GetPropertyAsync(string propertyName):Task<TypedProperty>
GetPropertyDefinitionsAsync():Task<IEnumerable<TypedProperty>>
SetPropertiesAsync(IEnumerable<Property> properties):Task
SetPropertyAsync(string propertyName, object value):Task
Config
GetConfig():Task<JObject>
PaymentService
CheckPaymentRequestStatusWptAsync(CheckPaymentRequestStatusWptRequest checkPaymentRequestStatusWptRequest):Task<PaymentRequestStatus>
CheckPaymentStatusAsync(CheckPaymentStatusRequest checkPaymentStatusRequest):Task<PaymentStatus>
CheckPaymentStatusRequestAsync(CheckPaymentRequestStatusRequest checkPaymentRequestStatusRequest):Task<string>
CreateInvoiceAsync(CreateInvoiceRequest createInvoiceRequest):Task

GetPaymentRequestAsync(GetPaymentRequests getPaymentRequests):Task<IEnumerable<PaymentRequestItem>> GetPaymentsAsync(GetPaymentsRequest getPaymentsRequest):Task<IEnumerable<Payment>> SavePaymentRequestAsync(SavePaymentRequest savePaymentRequest):Task<SavePaymentResult>
Navigation
GetHeader():Task<JObject> GetFooter():Task<JObject>

5.7. Servisi (SPA)

Servisi atbild par datu pieprasījumu loģiku, no kurienes tiek veikti visi pieprasījumi un atgriežas dati, lai tos varētu pēc tam attēlot.

NavigationService

Apraksts: Apvieno Header un Footer servisus, iekš sevis satur galapunktu mainīgos.

Konstruktors: `navigationService()`

HeaderService

Apraksts: Serviss, kas atbild par galvenes datu pieprasījumiem. Mantojas (extends) no NavigationService.

Konstruktors: `headerService()`

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
async callApi()	-	Promise<[]|void>	Pieprasa header api galapunktu
getData()	-	Object	Atgriež iepriekš pieprasītos datus

FooterService

Apraksts: Serviss, kas atbild par kājenes datu pieprasījumiem. Mantojas (extends) no NavigationService.

Konstruktors: `footerService()`

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
async callApi()	-	Promise<[]|void>	Pieprasa kājenes api galapunktu
handleData(data)	data - Object	Object	Apstrādā saņemtos datus

getData()	-	Object	Atgriež iepriekš pieprasītos datus

NotificationService

Apraksts: Serviss, kas atbild par notifikācijas datu pieprasījumiem.

Konstruktors: `NavigationService(options = {returnFullResponse: false}))`

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
async sendKdvNotifaction(data)	data - Object	Promise<object>	Nosūt kdv notifikāciju
async sendEmailNotification(data)	data - Object	Promise<object>	Nosūt email notifikāciju

PaymentService

Apraksts: Serviss, kas atbild par maksājuma datu pieprasījumiem.

Konstruktors: `PaymentService(options = {returnFullResponse: false}))`

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
async createPayment(data)	data - Object	Promise<object>	Izveido jauno maksājumu
async checkPaymentRequestData(data)	data - Object	Promise<object>	Atgriež maksājuma pieprasījuma statusu
async getPaymentRequests(data)	data - Object	Promise<object>	Atgriež maksājuma pieprasījumus
async getPayments(data)	data - Object	Promise<object>	Atgriež maksājumus
async checkPaymentStatus(data)	data - Object	Promise<object>	Pārbauda maksājuma statusu
async checkPaymentStatusWpt(data)	data - Object	Promise<object>	Pārbauda maksājuma wpt statusu
async createInvoice(data)	data - Object	Promise<object>	Izveido jauno rēķinu

EventService

Apraksts: Serviss, kas atbild par e-pakalpojuma notikumiem.

Konstruktors: `EventService()`

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
addEserviceStart(func)	Func – Function, atzvanišanas funkcija	-	Izveido notikumu kas būs izsaukts kad lietotājs sāks e-pakalpojumu
addEserviceCancel(func)	Func – Function, atzvanišanas funkcija	-	Izveido notikumu kas būs izsaukts kad lietotājs atcels e-pakalpojumu
addEserviceFinish(func)	Func – Function, atzvanišanas funkcija	-	Izveido notikumu kas būs izsaukts kad lietotājs pabeigs e-pakalpojumu
addEserviceSubmit(func)	Func – Function, atzvanišanas funkcija	-	Izveido notikumu kas būs izsaukts kad lietotājs iesniegs e-pakalpojuma soli
addEvent(name, func)	Name – String, notikuma nosaukums Func – Function, atzvanišanas funkcija	-	Izveido notikumu ar specifisko nosaukumu

AuthService

Apraksts: Serviss, kas atbild par autorizācijas pieprasījumiem.

Konstruktors: `AuthService()`

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
getUserItemsFromAuthList(authList, nameId)	authList – Object, user.profile.listofuauthentications vērtība kas atnāk no Ids servisa. nameId – String, user.profile.nameid	Object<{activeIndex: number, items: []}>	Atgriež lietotāju sarakstu ar aktīvo indeksi

	vērtība kas atnāk no Ids servisa.		
getAuthType(user)	user – Object, vērtība kas atnāk no Ids servisa (getUser)	String	Atgriež lietotāja tipu
getAuthTypeByRefId(refId)	refId – String, Object.entries(user.profile.listofuauthentications)[0].RefId vērtība kas atnāk no Ids servisa	String	Atgriež lietotāja tipu
getLegalTypeByRefId(refId)	refId – String, Object.entries(user.profile.listofuauthentications)[0].RefId vērtība kas atnāk no Ids servisa	String	Atgriež lietotāja tipu (legal type)
getUserIdentifierBySub(subKey, authType)	subKey – String, user.profile.nameid vērtība kas atnāk no Ids servisa; authType – String, vērtība kas atnāk no AuthService.getAuthType vai AuthService.getAuthTypeByRefId funkcijas;	String	Atgriež lietotāja identifikatoru (personas kodu)
checkAccountCanBeSelected(refId)	refId – String, Object.entries(user.profile.listofuauthentications)[0].RefId vērtība kas atnāk no Ids servisa	Promise<{msg: string, submsg: string, status: number}>	Atgriež objektu kur ir noradīts vai izvēletais lietotājs var būt izvēlēts.
changeUser(refId, successCallback, errorCallback)	refId – String, Object.entries(user.profile.listofuauthentications)[0].RefId; successCallback – Function, atskaņošanas funckija jā rezultāts bija veiksmīgs; errorCallback – Function, atskaņošanas funckija jā rezultāts bija neveiksmīgs;	-	Samaina aktīvo lietotāju uz citu (izvēlēto)

getGrantorType(grantorId)	grantorId – String, lietotāja identifikators	Number	Atgriež piešķīrēja tipu(Anonymoys, Authority, Company, vai Person)
getAuthTypeNumber(nameIdentifier)	nameIdentifier – String, lietotāja identifikators	Number	Atgriež piešķīrēja tipu (Company, Person, vai Anonymous)
getAllowedProviders(authType)	authType – Number, vērtība kas atnāk no AuthService. getAuthTypeNumber funkcijas	String[]	Atgriež sarakstu ar atļautiem autentifikācijas provaideriem
getAuthTypeMethodGroup(authMethod)	authMethod – String, user.profile.amr[0] vērtība kas atnāk no Ids servisa.	String|null	Atgriež provaidera pilno identifikatoru
checkAuthProvider(authMethodReference, allowedAuthMethods)	authMethodReference – String, user.profile.amr[0] vērtība kas atnāk no Ids servisa; allowedAuthMethods – String[],vērtība kas atnāk no AuthService. getAllowedProviders funkcijas	Boolean	Atgriež būla vērtību kas nozīme vai lietotājs var būt autentificēts
validateUser(user)	user – Object, user vērtība kas atnāk no Ids servisa.	Promise<void>	Izmēt kļūdu ja validācijas rezultāts bija neveiksmīgs.
decryptUtKey(key)	key – String, vērtība ko ir jadekriptē	Promise<{loginHint: string, authType: string}>	Atgriež dekriptēto vērtību “ut” parametram.
checkSession(pageRequiredAuth = true, forceAuth = false, redirectUri = “/”)	pageRequiredAuth – Boolean, vērtība kas pārbauda vai lietotājs ir autentificēts lai turpināt darbu;	Promise<{success: boolean, msg?: string, callback?: function}>	Atgriež sesijas validācijas rezultātu

	forceAuth – Boolean, vērtība kas norāda vai lietotāju ir automātiski jāautentificē		
getUser()	-	Object	Atgriež autentificēta lietotāja objektu
async getToken()	-	Promise<String>	Atgriež autentificēta lietotāja access_token
login()	-	-	Novirza lietotāju uz autentificēšanās lapu
renewToken()	-	-	Atjauno tokenu
logout()	-	-	Novirza lietotāju uz izlogošanās lapu

EService

Apraksts: Serviss, kas atbild par e-pakalpojumu pieprasījumiem. Visi pieprasījumi uz bff (backend-for-frontend) tiek veikti no šī servisa.

Konstruktors:

```
EService(props, options = {returnFullResponse: false})
```

- `props` – Object, objekts ar propertijiem
  - `props.history` – Object, react-router vēstures objekts

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
async start()	-	(Promise<void>|redirect)	Uzsāk e-pakalpojuma izpildi, novirza lietotāju uz pirmo soli, un atgriež saņemto e-pakalpojuma objektu
async get(routes, queryParams)	routes - String[], papildus maršrutētāji pieprasījumam queryParams - String, papildus	Promise<null|object>	Iegūst aktuālos(sāktā) e-pakalpojuma datus un atgriež tos

	query parametri pieprasījumam		
async cancel()	-	Promise<void>	Atceļ esošo e-pakalpojumu
async check()	-	Promise<void>	Pārbauda, vai SessionStorage ir saglabāts TransactionID - ja ir, tad izpilda metodi get()
async goto()	-	Promise<void>	Novirza lietotāju uz esošo e-pakalpojuma soļa lapu
async submit(data)	Data - Object, objekts ar datiem	Promise<object>	Iesniedz (submit) soli

TransactionService

Apraksts: Serviss, kas atbild par e-pakalpojuma transakcijām.

Konstruktors:

```
TransactionService(tabId, urn)
```

- `tabId` – String, lapas x-tabId
- `urn` – String, E-pakalpojuma URN

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
async endTransaction()	-	Promise<null|object>	Pabeidz transakciju

ProfileService

Apraksts: Serviss profila parametru pieprasījumiem.

Konstruktors:

```
ProfileService(options = {returnFullResponse: false})
```

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
async getAllProperties()	-	Promise<[]|void>	Atgriež visus profila parametrus ar to vērtībām
async getProfileProperty(name)	name - String	Promise<[]|void>	Atgriež profila parametru name

async getAllPropertyDefinitions()	-	Promise<[]|void>	Atgriež visas profila parametru definīcijas
async setProfileProperty(name, value)	name - String value - String	Promise<[]|void>	Uzstāda profila parametram name vērtību value

async setProfileProperties(data)	data – Object
	`data = {`
	`'applicationName': 'lvp',`
	`'properties': [`
	`{`
	`'name': 'userdefinedPropertyName',`
	`'value': 'userdefinedPropertyValue'`
	`}`
	`]`
	`}`
		Promise<[]|void>	Uzstāda definētajiem profila parametriem jaunās vērtības.
			Struktūra ir aprakstīta kolonnā "Parametri", kur "userdefined[..] " vērtības ir lietotāja definētās profila parametru jaunās vērtības;

SessionService

Apraksts: Serviss sesijas parametru pieprasījumiem.

Konstruktors:

```
SessionService(options = {returnFullResponse: false})
```

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
async getSessionAttributes()	-	Promise<null|object>	Atgriež visus sesijas atribūtus
async getSessionAttribute(name)	name - String	Promise<[]|void>	Atgriež sesijas atribūtu, kas padots caur parametru name
async setSessionAttribute(name, value)	name - String value - String	Promise<[]|void>	Uzstāda sesijas name atribūtam value vērtību

async deleteSessionAttribute(name)	name - String	Promise<[]|void>	Izdzēš sesijas name atribūtu

DocumentService

Apraksts: Serviss dokumentu apstrādes pieprasījumiem.

Konstruktors:

```
DocumentService(options = {returnFullResponse: false})
```

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
async getEdkDocuments()	-	Promise<[]|void>	Atgriež visus lietotāja dokumentus
async getEdkDocument(name)	name – String	Promise<[]|void>	Atgriež dokumentu ar cmis:objectId vērtību name
async getEdkDocumentContent(name)	name – String	Promise<[]|void>	Atgriež dokumenta datni, kur dokumenta cmis:objectId vērtība ir name
async createEdkDocument(propArray)	propArray – Array `propArray = [ { 'id': 'userSelectedDocumentId', 'value': 'userSelectedNewValueforProperty' } ]`	Promise<[]|void>	Izveido dokumentu ar lietotāja padotajām vērtībām. Dokumenta izveidei obligāts parametrs ir parametrs ar id “cmis:name”
async updateEdkDocument(docName, data)	docName - String data – Object `data = { 'properties': [ { 'id': 'userselectedPropertyId', 'value': 'userdefinedPropertyValue'`	Promise<[]|void>	Atjauno dokumenta, kura cmis:objectId ir docName, parametru vērtības, kas ir definētas iekš data objekta atslēgas 'properties'.

	<pre>}
	]
	
	}</pre>			

5.8. Mixins (SPA)

Palīg/kopējas funkcijas priekš e-pakalpojuma izstrādāšanas. Mixin\i sevī ietver visas palīgfunkcijas, kas ir nepieciešamas e-pakalpojuma darbībai, piemēram, pieprasījumi, SessionStorage funkcijas, sīkdatnes funkcionalitātes, u.c.

AssetsCdn

Apraksts: Datne ar funkcijām, kas palīdz strādāt ar assets_cdn

Exportējamās funkcijas:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
getAssetsCdnPath()	-	String	atgriež assets_cdn saiti

Cookies

Apraksts: Klase, kas atbild par darbībām ar sīkdatnēm (cookies)

Konstruktors:

`Cookies()`

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
set(key, value, days)	key - String, value - String, days - Number	-	Iestata sīkdatnes vērtību
get(key)	key - String	String	Atgriež sīkdatnes vērtību
remove(key)	key - String	-	Izdzēš sīkdatnes vērtību

FileStorage

Apraksts: Klase, kas atbild par darbībām ar datnēm - tiek izmantots iekš `Step.js` komponentes

Konstruktors:

`FileStorage(array)`

- `array` – Array, masīvs ar failu datiem

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
getStorage()	-	Array	Atgriež datnes datus

setStorage(arr)	arr – Array, jauno failu masīvs	-	Iestata jaunus datus
removeFile(file)	file – File, failu objekts	-	Izdzēš konkrēto failu no datņu masīva, kā identifikatoru izmantojot tas nosaukumu

i18n

Apraksts: Lokalizācijas klase, kas atbild par lietotnes tulkošanu.

Konstruktors:

`Internationalization(languages)`

- `languages` – String[], iespējamās valodas

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
getCurrentLanguage	-	String	Atgriež aktuālo izvēlēto valodu
updateTranslations(data)	data - Array	-	Atjauno saglabātos tulkojuma datus iekš localStorage
updateCurrentLanguage(lang)	lang - String	-	Atjauno izvēlēto valodu
isThereLangInString(str)	str - String	Boolean	Pārbauda, vai saņemtā vērtība sevī iekļauj kādu no valodas atslēgām
getLangFromPath(path)	path - String	String	Atgriež pirmos divus burtus no saņemtās vērtības
replaceKeysInString(str)	str - String	String	Samaina translation key šablonus (pattern) saņemtajā vērtībā uz vārdiem, un atgriež jauno
getTranslation(key, pattern)	key - String, pattern - Boolean	String	Atgriež tulkojumu saņemtajai atslēgai

request

Apraksts: Atbild par pieprasījumu veikšanu, izmantojot `axios` bibliotēku

Exportējamās funkcijas:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
async getRequestHeader()	-	Promise<{"x-tabId": string, "Authorization": (string|undefined)}>	Atgriež headers priekš jebkura pieprasījuma
async postData(endpoint, data, headers, checkAuthHeader)	endpoint - String, data - Object, headers - Object, checkAuthHeader - Boolean	Promise<string|Axios Response<any>>	Izpilda post pieprasījumu
async putData(endpoint, data, headers, checkAuthHeader)	endpoint - String, data - Object, headers - Object, checkAuthHeader - Boolean	Promise<string|Axios Response<any>>	Izpilda put pieprasījumu
async postFormData(endpoint, data, headers, checkAuthHeader)	endpoint - String, data - Object, headers - Object, checkAuthHeader - Boolean	Promise<string|Axios Response<any>>	Izpilda post pieprasījumu, kā padotos datus izmantojot formas iesūtīšanu caur XML pieprasījumu
async getData(endpoint, data, checkAuthHeader)	endpoint - String, data - Object, checkAuthHeader - Boolean	Promise<string|Axios Response<any>>	Izpilda get pieprasījumu
async deleteData(endpoint, data, headers, checkAuthHeader)	endpoint - String, headers - Object, checkAuthHeader - Boolean	Promise<string|Axios Response<any>>	Izpilda delete pieprasījumu

SessionStorage

Apraksts: Klase, kas atbild par darbībām ar sessionStorage browser api.

Konstruktors:

`SessionStorage()`

Metodes:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
set(key, value)	key - String, value - Any	-	Iestata sesijas vērtību
get(key)	key - String	Any	Atgriež sesijas vērtību
remove(key)	key - String	-	Izdzēš sesijas vērtību

clear()	-	-	Izdzēš visas sesijas vērtības

Typer

Apraksts: Atbild par pareizo datu tipu konvertēšanu.

Eksportējamas funkcijas:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
getCorrectTypedValue(value)	value - String	Any	Pārveido saņemto virkni par vērtību ar pareizo tipu
getCorrectTypedProps()	origObj - Object	Object	Iterē cauri visam objektam un pārveido tās vērtības pareizajos tipos (deeply)

IsDefinedInStorage

Apraksts: Atbild par mainīgo pārbaudīšanu iekš ConfigStore.

Eksportējamas funkcijas:

Nosaukums	Parametri	Atgriežamā vērtība	Apraksts
async IsDefinedInStoreAsync(value)	value - String	Promise<boolean|string>	Pārbauda, vai vērtība ir nodefinēta iekš ConfigStore
IsDefinedInStore(value)	value - String	Boolean	Pārbauda, vai vērtība ir nodefinēta iekš ConfigStore

5.9. MVC(MPA) palīgmetodes

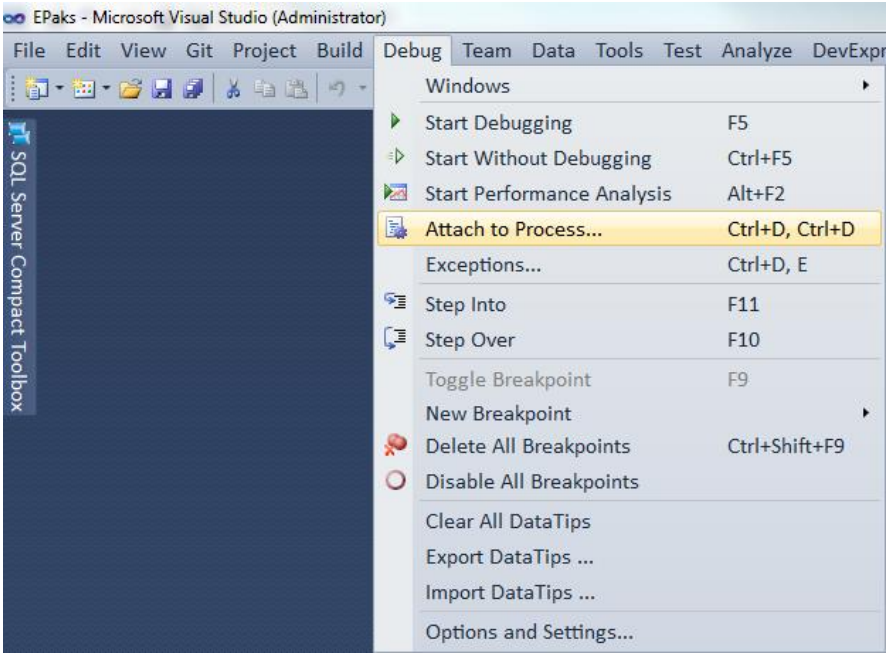
MPA pieejas skata modelī pieejamas šādas palīgmetodes

Metode	Izsaukums	Rezultāts	Piezīmes
LocalizeUrl	LocalizeUrl(url) • url(obligāts) – simbolu virkne, saite ar valodas vietturi ({language}), kuru jāapstrādā	Simbolu virkne	Aizvieto valodas vietturi ar lietotāja izvēlēto valodu

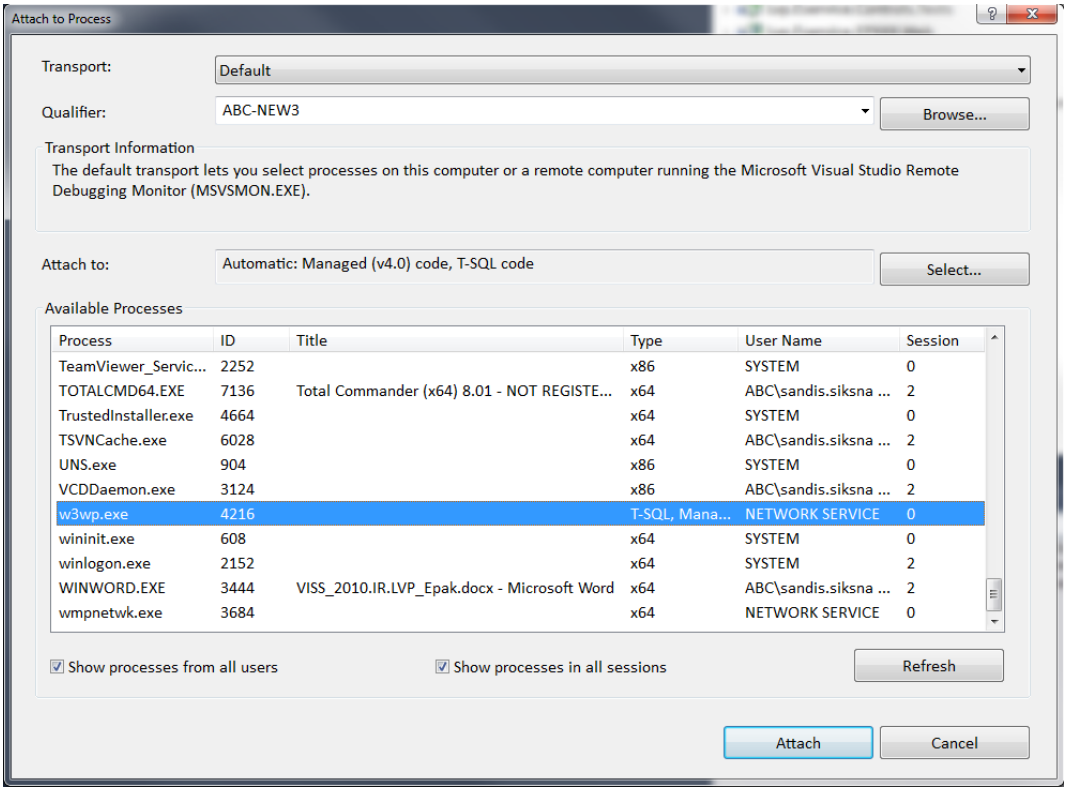
GetEServiceId	GetEServiceId(urn) • urn(obligāts) – simbolu virkne, e-pakalpojuma urn	Simbolu virkne	Parsē e-pakpalpojuma URN lai izgūtu e-pakalpojuma id

5.10.IIS darbināta projekta atkļūdošana (debug)

IIS darbinātu tīmekļa lietojumu var atkļūdot, pieslēdzoties *w3wp.exe* procesam:



3.attēls. Pieslēgšanās dialoga atvēršana



4.attēls. Pieslēgšanās IIS procesam

6. Bibliotēkas e-pakalpojumu izstrādei

Atkarībā no izvēlētās pieejas e-pakalpojumu klienta puses izstrādei ir pieejamas šādas gatavas bibliotēkas, kuras atbalsta dizaina vadlīnijas:

- ReactSDK – React bibliotēka ietver sevī gatavas react komponentes priekš e-pakalpojumu izstrādes SPA režīmā. ReactSDK dokumentācija pieejama https://eservices-test.vraa.gov.lv/EservicePlatform.Controls.React/ . ReactSDK bibliotēku e-pakalpojumos var pievienot divos veidos:
  - Asset cdn – Bibliotēku iespējams pieprasīt un izmantot no CDN, tā atrodas šādā adresē – <CDN-Adrese>/<Versija>/SDK/ReactSDK/js/react-sdk.min.js. Attiecīgās bibliotēkas stili atrodās šādā adresē - <CDN-Adrese>/<Versija>/SDK/ReactSDK/css/style.css. Lietojumu var skatīt https://eservices-test.vraa.gov.lv/EservicePlatform.Examples.React.ComplexUI/ piemēros
  - Npm – jāpieslēdz repozitorijs, kurā atrodas @eserviceplatform/controls-react un pie npm konfigurācijas jānorāda, nepieciešamā versija. Šajā gadījumā nebūs iespējams dinamiski mainīt bibliotēkas versiju
- HTMLSDK – Javascript bibliotēka, kura izstrādāta specifiski priekš šī projekta dizaina komponentēm, to var izmantot bez papildus bibliotēkām. Bibliotēku paredzēts izmantot, lai piešķirtu funkcionalitāti HTML elementiem. HTMLSDK dokumentācija pieejama šeit: https://eservices-test.vraa.gov.lv/EservicePlatform.Controls.Html/ . HTMLSDK iespējams iegūt divos veidos:
  - Asset cdn – Bibliotēku iespējams pieprasīt un izmantot no CDN, tā atrodās šādā adresē – <CDN-Adrese>/<Versija>/SDK/HTMLSDK/js/bundle.js. Attiecīgās bibliotēkas stili atrodas šādā adresē - <CDN-Adrese>/<Versija>/SDK/HTMLSDK/css/style.css.
  - Npm – jāpieslēdz repozitorijs, kurā atrodas @eserviceplatform/controls-html un pie npm konfigurācijas jānorāda, nepieciešamā versija. Šajā gadījumā nebūs iespējams dinamiski mainīt bibliotēkas versiju.
- MVC helpers – Izvēloties izstrādāt klienta pusi pēc MPA pieejas un izmantojot.Net Core MVC 3.1 ir iespējams lejupielādēt nuget bibliotēku - Lvp.EservicePlatform.Controls.Mvc, kura sevī iekļauj gatavas komponentes, kuras veidotas uz HTMLSDK bāzes. Lai nodrošinātu dizainu un JavaScript funkcionalitāti komponentēm ir jāiekļauj attiecīgās versijas HTMLSDK css stili un JavaSript bibliotēka, kuras pieejamas assetos(skat. HTMLSDK aprakstu). MVC helpers dokumentācija pieejama šeit: https://eservices-test.vraa.gov.lv/EservicePlatform.Controls.Mvc/doc/index.html
- React epakalpojumu izstrādei ir jāizmanto npm paciņa frontend-react, kura nodrošina e-pakalpojuma bāzes darbības un dizaina integrāciju līdz tādam līmeni, ka izstrādātājam ir jānorāda tikai konfigurācija un e-pakalpojuma biznesa loģika. Pieejamās palīgmetodes aprakstītas 5. punktā.
- MVC un React starpslāņa izstrādei jāizmanto EServiceCore Nuget Pakotne - Lvp.EservicePlatform.Backend.Essentials.EserviceCore. Kura ir pieejama VRAA -nuget repozitorijā. Pakotne nodrošina epakalpojuma izpildes algoritmu, LVPContext servisus un Dizaina integrāciju MPA variantam. Izstrādātājam ir jānorāda tikai konfigurācija un jāizstrādā e-pakalpojuma biznesa loģika.

6.1. Notikumu žurnalēšana

Notikumu žurnalēšana tiek nodrošināta, izmantojot pielāgotu standarta *.NET Core* žurnalēšanas abstrakcijas (`Microsoft.Extensions.Logging.ILogger<T>`) *Serilog* realizāciju. Lai pievienotu notikumu žurnalēšanu, nepieciešams:

- pievienot NUGET pakotnes
  - `Abc.Analytics.Serilog`
  - `Abc.Analytics.Serilog.AspNetCore`

Detalizētāku informāciju skatīt dokumenta [9] nodaļā “5. Žurnalēšanai no konteinerizētām komponentēm”.

7. Servisi e-pakalpojumu izstrādei

7.1. LvpContext.SessionProperties

Šajā nodaļā aprakstītas metodes e-pakalpojuma sadarbībai ar sesijas servisu.

Dažas sesijas īpašības (piemēram, `transactionId`, `nameidentifier`) ir sistēmas noteiktas (*read only*) – e-pakalpojums tās nevar mainīt. Sesijas īpašību identifikatori nav reģistrjūtīgi. Maksimālais sesijas īpašības identifikatora garums – 100 simboli, bet vērtības – 2000 simboli.

7.1.1. Sesijas īpašību izgūšana

IDENTIFIKATORS	LvpContext.SessionProperties
APRAKSTS	Izgūst lietotāja (gan anonīma, gan autentificēta) aktuālajā e-pakalpojuma sesijā esošās īpašības.

METODES IZSAUKŠANA

Adrese

`GET EservicePlatform.ContextAPI/api/v1/sessionproperties`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons (autentificētiem izsaukumiem) vai nenorādīts (anonīms izsaukums).
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

Nav.

QUERY parametri

Nav.

BODY parametri

Nav.

Piemērs

```
GET EservicePlatform.ContextAPI/api/v1/sessionproperties HTTP/1.1

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377
```

IZVADDATI

Lietotāja (gan anonīma, gan autentificēta) aktuālajā e-pakalpojuma sesijā esošo īpašību un to vērtību kolekcija (vārdnīcas veidā). Visu sesijas īpašību vērtību tips ir simbolu virkne (*string*).

Kļūdas

- `HTTP 401` – norādīts nederīgs autentifikācijas talons.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{"nameidentifier":"PK:11111111111","transactionid":"URN:IVIS:100001:EF.VISS-
EF00-v1-0-TR-635"}
```

7.1.2. Sesijas īpašības vērtības izgūšana

IDENTIFIKATORS	LvpContext.SessionProperty
APRAKSTS	Izgūst lietotāja (gan anonīma, gan autentificēta) aktuālajā e-pakalpojuma sesijā esošas īpašības vērtību.

METODES IZSAUKŠANA

Adrese

`GET EservicePlatform.ContextAPI/api/v1/sessionproperties/{propertyId}`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons (autentificētiem izsaukumiem) vai nenorādīts (anonīms izsaukums).
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametriH

- `propertyId` – sesijas īpašības identifikators.

QUERY parametri

Nav.

BODY parametri

Nav.

Piemērs

```
GET EservicePlatform.ContextAPI/api/v1/sessionproperties/property1 HTTP/1.1

x-tabId: 1233ff35bd234e3082d3e994da33c377
```

IZVADDATI

Sesijas īpašības vērtība kā simbolu virkne (*string*) vai `HTTP 204`, ja sesijas īpašība neeksistē, t.i., tai nav vērtības.

Kļūdas

- `HTTP 401` – norādīts nederīgs autentifikācijas talons.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=utf-8

PK:11111111111
```

7.1.3. Sesijas īpašības vērtības aktualizēšana

IDENTIFIKATORS	LvpContext.SetSessionProperty
APRAKSTS	Aktualizē norādītās sesijas īpašības vērtību.

METODES IZSAUKŠANA

Adrese

`PUT EservicePlatform.ContextAPI/api/v1/sessionproperties/{propertyId}`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons (autentificētiem izsaukumiem) vai nenorādīts (anonīms izsaukums).
- `Content-Type` - `application/json; charset=utf-8`.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

- `propertyId` – sesijas īpašības identifikators.

QUERY parametri

Nav.

BODY parametri

- `value` – uzstādāmās īpašības vērtība. Lai dzēstu vērtību, kā vērtība jānorāda `JSON NULL`, t.i., "`{ "value": null }`".

Piemērs

```
PUT EservicePlatform.ContextAPI/api/v1/sessionproperties/property123 HTTP/1.1

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377
Content-Type: application/json; charset=utf-8

{
"value":"jauna sesijas īpašības vērtība"
}
```

IZVADDATI

`HTTP 204` vai kļūda.

Kļūdas

- `HTTP 401` – norādīts nederīgs autentifikācijas talons.
- `HTTP 400` – norādītās sesijas īpašības identifikators pārsniedz maksimāli atļauto garumu.
- `HTTP 400` – norādītās sesijas īpašības vērtība pārsniedz maksimāli atļauto garumu.
- `HTTP 400` – norādītās sesijas īpašības identifikators atbilst sistēmas noteiktai īpašībai.

Piemērs

```
HTTP/1.1 204 No Content
```

7.1.4. Sesijas īpašības vērtības dzēšana

IDENTIFIKATORS	LvpContext.DeleteSessionProperty
APRAKSTS	Dzēš norādītās sesijas īpašības vērtību.

METODES IZSAUKŠANA

Adrese

`DELETE EservicePlatform.ContextAPI/api/v1/sessionproperties{propertyId}`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons (autentificētiem izsaukumiem) vai nenorādīts (anonīms izsaukums).
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

- `propertyId` – dzēšamās sesijas īpašības identifikators. Nav iespējams dzēst vērtību sistēmas noteiktām īpašībām.

QUERY parametri

Nav.

BODY parametri

Nav.

Piemērs

```
DELETE EservicePlatform.ContextAPI/api/v1/sessionproperties/property123 HTTP/1.1

x-tabId: 0643ff35bd234e3082d3e994da33c377
```

IZVADDATI

`HTTP 204` vai kļūda.

Kļūdas

- `HTTP 401` – norādīts nederīgs autentifikācijas talons.
- `HTTP 400` – norādītās sesijas īpašības identifikators atbilst sistēmas noteiktai īpašībai.

Piemērs

```
HTTP/1.1 204 No Content
```

7.1.5. Vairāku sesijas īpašību vērtību aktualizēšana

IDENTIFIKATORS	LvpContext.SetSessionProperties
APRAKSTS	Aktualizē norādīto sesijas īpašību vērtības.

METODES IZSAUKŠANA

Adrese

`PUT EservicePlatform.ContextAPI/api/v1/sessionproperties`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons (autentificētiem izsaukumiem) vai nenorādīts (anonīms izsaukums).
- `Content-Type` - `application/json; charset=utf-8`.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

Nav.

QUERY parametri

Nav.

BODY parametri

- `properties` - uzstādāmo īpašības saraksts, obligāts. Sarakstam jāsatur vismaz viena uzstādāmā īpašība. Sarakstā nedrīkst būt sistēmas noteiktās īpašības (piemēram, transactionId).Katrai sarakstā esošajai īpašībai ir `name` un `value` atribūti:
  - `name` - īpašības nosaukums (nav reģistrjūtīgs), obligāts;
  - `value` - uzstādāmās īpašības vērtība. Īpašību vērtības jānorāda `JSON String` formātā. Lai dzēstu vērtību, ir iespējams uzstādīt `JSON NULL`.

Piemērs

```
PUT EservicePlatform.ContextAPI/api/v1/sessionproperties HTTP/1.1

x-tabId: 0643ff35bd234e3082d3e994da33c377

{
  "properties": [
  {
    "name":"ipasiba1",
    "value":"jaunavertiba1"
  },
  {
    "name":"ipasiba2",
    "value":"jaunavertiba2"
  }
  ]
}
```

IZVADDATI

`HTTP 204` vai kļūda.

Kļūdas

- `HTTP 401` – norādīts nederīgs autentifikācijas talons.
- `HTTP 400` – nekorekti pieprasījuma dati – piemēram, kādas norādītās sesijas īpašības identifikators atbilst sistēmas noteiktai īpašībai; nenorādīts vai norādīts tukšs uzstādāmo īpašības saraksts.

Piemērs

```
HTTP/1.1 204 No Content
```

7.1.6. Visu sesijas īpašību vērtību dzēšana

IDENTIFIKATORS	LvpContext.DeleteSessionProperties
APRAKSTS	Dzēš visas (izņemot sistēmas noteiktās, piemēram, transakcija) lietotāja sesijas īpašību vērtības.

METODES IZSAUKŠANA

Adrese

`DELETE EservicePlatform.ContextAPI/api/v1/sessionproperties`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons (autentificētiem izsaukumiem) vai nenorādīts (anonīms izsaukums).
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

Nav.

QUERY parametri

Nav.

BODY parametri

Nav.

Piemērs

```
DELETE EservicePlatform.ContextAPI/api/v1/sessionproperties HTTP/1.1

x-tabId: 0643ff35bd234e3082d3e994da33c377
```

IZVADDATI

`HTTP 204` vai kļūda.

Kļūdas

- `HTTP 401` – norādīts nederīgs autentifikācijas talons.

Piemērs

```
HTTP/1.1 204 No Content
```

7.2. LvpContext.Request

Šajā nodaļā aprakstītas metodes e-pakalpojuma transakcijas uzsākšanai un noslēgšanai.

Visas apakšnodaļā aprakstītās metodes ir iespējams izsaukt autentificētā veidā. Lai izsauktu autentificētā veidā nepieciešams *LVP.IdentityProvider* izsniegts un izpildes laikā derīgs *OAuth2* (*JWT* vai *references*) talons. Daļu no metodēm ir iespējams izsaukt arī anonīmā veidā, tas ir norādīts katras metodes aprakstā.

7.2.1. Jaunas e-pakalpojuma transakcijas izveide

IDENTIFIKATORS	LvpContext.RequestService.CreateTransaction
APRAKSTS	Jaunas e-pakalpojuma transakcijas izveide un uzsākšana. Šo metodi ir iespējams izsaukt arī anonīmā veidā. ***Izveidota transakcija tiek saglabāta attiecīgā lietotāja sesijā (skat. 7.1. nodaļa) un infrastruktūra to izmanto servisu, kuros ir nepieciešama transakcija, pieprasījumos.***

METODES IZSAUKŠANA

Adrese

`POST EservicePlatform.ContextAPI/api/v1/request/transactions`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons (autentificētiem izsaukumiem) vai nenorādīts (anonīms izsaukums).

- `Content-Type` - `application/json; charset=utf-8`.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

Nav.

QUERY parametri

Nav.

BODY parametri

- `eServiceId` (*string*) – e-pakalpojuma, kuram nepieciešams veidot transakciju, identifikators (*URN*).

Piemērs

```
POST EservicePlatform.ContextAPI/api/v1/request/transactions HTTP/1.1

Content-Type: application/json; charset=utf-8
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377

{
"eServiceId": "URN:IVIS:100001:EF.VISS-EF00-v1-0"
}
```

IZVADDATI

Izveidotās e-pakalpojuma izpildes transakcijas identifikators (*URN*) vai `HTTP 204`, ja transakcija netika izveidota (piemēram, nepareiza e-pakalpojuma identifikatora dēļ).

Kļūdas

- `HTTP 401` - norādīts nederīgs autentifikācijas talons.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=utf-8

URN:IVIS:100001:EF.VISS-EF00-v1-0-TR-43
```

7.2.2. E-pakalpojuma transakcijas apturēšana

IDENTIFIKATORS	LvpContext.RequestService.StopTransaction
APRAKSTS	E-pakalpojuma transakcijas apturēšana. Šo metodi ir iespējams izsaukt arī anonīmā veidā.

METODES IZSAUKŠANA

Adrese

`DELETE EservicePlatform.ContextAPI/api/v1/request/transactions/{transactionId}`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons (autentificētiem izsaukumiem) vai nenorādīts (anonīms izsaukums).
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

- `transactionId` – e-pakalpojuma transakcijas identifikators (*URN*), kuru ir nepieciešams apturēt.

QUERY parametri

Nav.

BODY parametri

Nav.

Piemērs

```
DELETE
EservicePlatform.ContextAPI/api/v1/request/transactions/URN%3AIVIS%3A100001%3AE
F.VISS-EF00-v1-0-TR-42 HTTP/1.1

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377
```

IZVADDATI

Veiksmīgas e-pakalpojuma transakcijas apturēšanas gadījumā tiek izvadīts `JSON TRUE`, neviksmīgas – `JSON FALSE`.

Kļūdas

- `HTTP 401` - norādīts nederīgs autentifikācijas talons.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

true
```

7.2.3. Integrācijas servisu izsaukšana

IDENTIFIKATORS	LvpContext.RequestService.IvisRequest
APRAKSTS	Nodrošina API Pārvaldniekā reģistrētu (SOAP) integrācijas servisu izsaukšanu (detalizētāk skat. [6]), nodrošinot IVISRequest struktūras izveidi un aizpildīšanu. ***Šī metode nodrošina iespēju izsaukt SOAP integrācijas servisus e-pakalpojuma izstrādes videi neatkarīgā veidā, neveicot izmaiņas integrācijas servisā.*** ***Lai izpildītu šo metodi, nepieciešama aktīva e-pakalpojuma transakcija attiecīgā lietotāja sesijā.***

METODES IZSAUKŠANA

Adrese

```
POST
EservicePlatform.ContextAPI/api/v1.0/request/ivisrequests?targetUrl={targetUrl}&mes
sageType={messageType}
```

HEADER parametri

- `Authorization` – *Bearer OAuth2* (*JWT* vai *references*) talons, obligāts.
- `Content-Type` – `application/xml; charset=utf-8`, obligāts.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.

- `x-milestoneId` – e-pakalpojuma izpildes robežpunkta identifikators, obligāts.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/xml;charset=utf-8`, `application/json`, `plain/text`.

PATH parametri

Nav.

QUERY parametri

- `targetUrl` – izsaucamā integrācijas servisa relatīvā adrese Url encoded formātā, obligāts.
- `messageType` – `IVISRequest` ziņojuma tips, obligāts.

BODY parametri

- `messageBody` – `IVISRequest` ziņojuma saturs (*body*) `XML` formātā, obligāts.

Piemērs

```
POST EservicePlatform.ContextAPI/api/v1.0/request/ivisrequests?targetUrl=/ISS-
SIA.ABC-
CalculationDataSync/v1.0/CalculationSync.svc&messageType=URN:IVIS:100001:XSD-
Testing-TestISServise-v1-0-TYPE-Calculation HTTP/1.1
Content-Type: application/xml; charset=utf-8
x-tabId: 29A752A065DB4C2686C186C8CBF8303A
x-milestoneId: URN:IVIS:100001:EF.VISS-EF00-v1-0-MS-CallCalcSync
Authorization: Bearer j3ScL18D4Dnjll0np8cwT7T-mnQ9u3w4iriWyG2D4ts

<Calculation xmlns="http://ivis.eps.gov.lv/XMLSchemas/100000/TestISServise/v1-
0"             xmlns:ivis="http://ivis.eps.gov.lv/XMLSchemas/100001/IVIS/v1-0"
xmlns:pers="http://ivis.eps.gov.lv/XMLSchemas/100001/Person/v1-0">
<Number1>5</Number1>
<Number2>6</Number2>
<Operation>multiplication</Operation>
</Calculation>
```

IZVADDATI

Veiksmīga integrācijas servisa izsaukuma gadījumā tiek izvadīts `HTTP 200` un `IVISResponse` struktūra `XML` formātā ar pazīmi `success` vai `failure` (ja notikusi apstrādāta biznesa kļūda). Neviksmīga izsaukuma gadījumā tiks izvadīta kļūda.

`LvpContext.Request` neapstrādā gala resursa (Api Pārvaldnieka, integrācijas servisa) izvadītās atbildes, kuras nav `IVISResponse`, tās tiek izvadītas tieši tādas, kā tika saņemtas.

Kļūdas

- `HTTP 401` - norādīts nederīgs autentifikācijas talons.
- `HTTP 400` – nav norādīts vai norādīts nekorekts e-pakalpojuma izpildes robežpunkta identifikators.
- `HTTP 400` – nav norādīts vai norādīts nekorekts `IVISRequest` ziņojuma tips.
- `HTTP 400` – nav norādīta izsaucamā integrācijas servisa relatīvā adrese.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: application/xml; charset=utf-16

<?xml version="1.0" encoding="utf-16"?>
<ivis:IVISResponse
xmlns:pers="http://ivis.eps.gov.lv/XMLSchemas/100001/Person/v1-0"
xmlns:ivis="http://ivis.eps.gov.lv/XMLSchemas/100001/IVIS/v1-0">
  <ivis:Header>
    <ivis:MessageID>2abc1a34-9095-4be7-b911-b04c83008318</ivis:MessageID>
    <ivis:MessageType>URN:IVIS:100001:XSD-Testing-TestISServise-v1-0-TYPE-
Result</ivis:MessageType>
    <ivis:TransactionID>URN:IVIS:100001:EF.VISS-EF00-v1-0-TR-
878</ivis:TransactionID>
    <ivis:CorrelationID>1f441e8f-d326-4a7e-b750-
e1f391358219</ivis:CorrelationID>
    <ivis:TimeStamp>2020-05-08T12:55:48.5151022+03:00</ivis:TimeStamp>
    <ivis:Result>success</ivis:Result>
  </ivis:Header>
  <ivis:Body>
    <Result   xmlns="http://ivis.eps.gov.lv/XMLSchemas/100000/TestISServise/v1-
0"             xmlns:ivis="http://ivis.eps.gov.lv/XMLSchemas/100001/IVIS/v1-0"
xmlns:pers="http://ivis.eps.gov.lv/XMLSchemas/100001/Person/v1-0">
      <CalculationValue>30</CalculationValue>
    </Result>
  </ivis:Body>
</ivis:IVISResponse>
```

7.2.4. API servisu izsaukšana

IDENTIFIKATORS	LvpContext.RequestService.ApiRequest
APRAKSTS	Nodrošina Api Pārvaldniekā reģistrētu (*REST*) servisu izsaukšanu (detalizētāk skat. [6]). Šī metode darbojas kā starpnieks (*proxy*) – tā pārsūta visus saņemtos parametrus un saturu (t. sk. *HTTP* metodi, *HEADER*, *PATH*, *QUERY* un *BODY* parametrus) norādītajam API Pārvaldnieka servisam (`targetUrl`). Tiek pārsūtīti tikai atļauti (*whitelisted*) *HEADER* parametri. Šīs metodes aprakstā ir minēti tie parametri, kuri ir nepieciešami pašas metodes darbībai. ***Lai izpildītu šo metodi, nepieciešama aktīva e-pakalpojuma transakcija attiecīgā lietotāja sesijā.***

METODES IZSAUKŠANA

Adrese

`HTTP_METODE`
`EservicePlatform.ContextAPI/api/v1.0/request/apirequests?targetUrl={targetUrl}`

HEADER parametri

- `Authorization` – *Bearer OAuth2* (*JWT* vai *references*) talons, obligāts.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `x-milestoneId` – e-pakalpojuma izpildes robežpunkta identifikators, obligāts.
- `Accept` – vēlamais atbildes formāts. Atkarīgs no tā, ko nodrošina gala serviss.

PATH parametri

Nav.

QUERY parametri

- `targetUrl` – izsaucamā *API* servisa relatīvā adrese Url encoded formātā, obligāts.

BODY parametri

Nav.

Piemērs

```
POST
EservicePlatform.ContextAPI/api/v1.0/request/apirequests?targetUrl=/VISS.ApiMan
agement/Extensions/CalculationApi/stable/v1/api/Calculator/Divide HTTP/1.1
Content-Type: application/json; charset=utf-8
x-tabId: 29A752A065DB4C2686C186C8CBF8303A
x-milestoneId: URN:IVIS:100001:EF.VISS-EF00-v1-0-MS-CallCalcSync
Authorization: Bearer bN3duV-RsF9j7Icj6TadSBbeA2PiMQznJyalVeeG90c

{
  "Dividend": 15,
  "Divisor": 2
}
```

IZVADDATI

No API Pārvaldnieka vai tājā reģistrētā servisa saņemtā atbilde vai kļūda.

Kļūdas

- `HTTP 401` - norādīts nederīgs autentifikācijas talons.
- `HTTP 400` – nav norādīts vai norādīts nekorekts e-pakalpojuma izpildes robežpunkta identifikators.
- `HTTP 400` – nav norādīta izsaucamā integrācijas servisa relatīvā adrese.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

7.5
```

7.3. LvpContext.Edk

Šajā nodaļā aprakstītas metodes e-pakalpojuma sadarībai ar *EDK [5]*.

Visas sadaļā aprakstītās metodes ir iespējams izsaukt tikai autentificētā veidā, izmantojot *LVP.IdentityProvider* izsniegtu un izpildes laikā derīgu *OAuth2* (*JWT* vai *references*) talonu.

Visu metožu izpildei nepieciešama aktīva e-pakalpojuma izpildes transakcija, kuru var izveidot, izmantojot *LvpContext.Request* metodi (skat. 7.2.1. sadaļu), attiecīgā lietotāja sesijā.

2.tabula

EDK īpašību vērtību JSON datu tipi

EDK ĪPAŠĪBAS TIPS	JSON DATU TIPS	PIEZĪMES
PropertyId	JSON String	
PropertyString	JSON String	
PropertyDecimal	JSON String	
PropertyUri	JSON String	
PropertyHtml	JSON String	

EDK ĪPAŠĪBAS TIPS	JSON DATU TIPS	PIEZĪMES
`PropertyDateTime`	`JSON String`	Ievaddatos atļautie formāti: "*yyyy-MM-ddTHH:mm:ss.fffZ*", "*yyyy-MM-ddTHH:mm:ssZ*", "*yyyy-MM-ddZ*". Izvadē tiek izvadīts "*yyyy-MM-ddTHH:mm:ss.fffZ*".
`PropertyInteger`	`JSON Number`	
`PropertyBoolean`	`JSON True` vai `JSON False`	

7.3.1. Dokumenta izgūšana

IDENTIFIKATORS	LvpContext.EdkService.GetDocument
APRAKSTS	Izgūst gan autentificētā lietotāja aktuālās e-pakalpojuma sesijas transakcijā, gan citur (piemēram, lietotāja iepriekšējo transakcijās) esoša EDK dokumenta aprakstošo informāciju (tikai metadatus, t.i., īpašību sarakstu atbilstoši objekta tipa definīcijai). Ja tiek izgūts EDK dokumenta, kas atrodas ārpus autentificētā lietotāja aktuālās vai vēsturiskajām transakcijām, informācija, piemēram, EDK dokumentu veido kāda cita sistēma, tad šīs sistēmas EDK integrācijas procesā ir nepieciešams nodrošināt to, ka autentificētājam lietotājam attiecīgajam EDK dokumentam ir piešķirta vismaz `edk:getProperties` tiesība). *EDK* dokumentu tipu definīcijas pieejamas [5] dokumenta "5.3. EDK tipi" nodaļā.

METODES IZSAUKŠANA

Adrese

`GET EservicePlatform.ContextAPI/api/v1/edk/documents/{dokumentId}?filter={īpašību filtrs}`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

- `dokumentId` - dokumenta identifikators jeb `cmis:objectId`

QUERY parametri

- `filter` – dokumenta īpašību filtrs. Pēc noklusējuma dokumentam tiek izvadīta tikai `cmis:objectId` īpašība. Lai izvadītu visas īpašības (kuras ietilpst dokumenta objekta tipa definīcijā un kurām attiecīgajam dokumentam eksistē vērtība), kā filtrs jānorāda "`*`". Lai izvadītu konkrētas īpašības, tās jānorāda filtrā, atdalītas ar komatu, piemēram, "`cmis:objectId,cmis:name,cmis:description,cmis:objectTypeId`". "`cmis:objectId`" tiek izvadīta vienmēr (arī, ja neietilpst filtrā). Ja īpašību filtrā norāda īpašību, kas neeksistē attiecīgajam dokumentam (piemēram"`cmis:name123`") vai tā tipam, tā tiek ignorēta (nenotiek kļūda). Īpašību filtrs nav reģistrjūtīgs (*case-insensitive*) - "`cmis:name`" ir tas pats, kas "`CMIS:name`".

BODY parametri

Nav.

Piemērs

```
GET       EservicePlatform.ContextAPI/api/v1/edk/documents/URN:IVIS:200266:DOC-
1000095734-V1.0?filter=cmis:name HTTP/1.1

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377
```

IZVADDATI

Dokumenta īpašību saraksts. Dokumenta īpašību vērtības tiek izvadītas atbilstoši to īpašību tipiem skat. 2. tabulu.

Kļūdas

- `HTTP 404` - dokuments neeksistē *EDK*; dokuments eksistē, bet tas neietilpst aktuālās transakcijas mapes hierarhijā un lietotājam nav `edk:getProperties` tiesība.
- `HTTP 400` - dokumenta identifikators neatbilst *EDK* dokumenta identifikatora (*URN*) formātam.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

[{"id":"cmis:objectId","value":["URN:IVIS:200266:DOC-1000095299-V1.0"]}]
```

7.3.2. Dokumentu saraksta izgūšana

IDENTIFIKATORS	LvpContext.EdkService.GetDocuments
APRAKSTS	Izgūst autentificētā lietotāja aktuālās e-pakalpojuma sesijas transakcijā esošu *EDK* dokumentu, kuru tips ir vai manto `edk:d:customDocument` tipu, aprakstošo informāciju, t.i., tikai metadatus, t.i., īpašību sarakstu atbilstoši objekta tipa definīcijai kā sarakstu. *EDK* dokumentu tipu definīcijas pieejamas [5] dokumenta "5.3. EDK tipi" nodaļā. ***Tiek izvadītas tikai īpašības, kuras ietilpst `edk:d:customDocument` tipa definīcijā, t.i., ja dokumentam ir tips, kas manto šo tipu un dokumenta tipam ir savas īpašības, tad tās šajā metodē netiks izvadītas arī tādā gadījumā, ja tiks norādītas īpašību filtrā. Tāda veida īpašības ir iespējams izgūt, izgūstot individuālu EDK dokumentu (skat. 7.3.1. sadaļu), nevis dokumentu sarakstu.***

METODES IZSAUKŠANA

Adrese

`GET /api/v1/edk/documents?filter={īpašību filtrs}&skip=0&take=10`

HEADER parametri

- `Authorization` - Bearer *OAuth2* (*JWT* vai *references*) talons.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

Nav.

QUERY parametri

- `filter` – dokumentu īpašību filtrs. Pēc noklusējuma visiem izvadītajiem dokumentiem tiek izvadīta tikai `cmis:objectId` īpašība. Lai izvadītu visas īpašības (kuras ietilpst `edk:d:customDocument` definīcijā), kā filtrs jānorāda "`*`". Lai izvadītu konkrētas īpašības, tās

jānorāda filtrā, atdalītas ar komatu, piemēram, "cmis:objectId,cmis:name,cmis:description,cmis:objectTypeId". "cmis:objectId" tiek izvadīta vienmēr (arī, ja neietilpst norādītajā filtrā).

- skip – izlaižamo dokumentu skaits.
- take – izgūstamo dokumentu skaits.

BODY parametri

Nav.

Piemērs

```
GET /api/v1/edk/documents?filter=cmis:name&skip=0&take=10 HTTP/1.1

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377
```

IZVADDATI

Saraksts ar autentificētā lietotāja aktuālās sesijas transakcijā ietilpstošo dokumentiem (to īpašību sarakti). Dokumentu īpašību vērtības tiek izvadītas atbilstoši to īpašību tipiem skat. 2. tabulu.

Kļūdas

- HTTP 400 - norādīts skip, un tas nav vesels, nenegatīvs skaitlis.
- HTTP 400 - norādīts take, un tas nav vesels, nenegatīvs skaitlis.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

[[{"id":"cmis:objectId","value":["URN:IVIS:200266:DOC-1000095501-V1.0"]}]]
```

7.3.3. Jauna dokumenta izveidošana

IDENTIFIKATORS	LvpContext.EdkService.CreateDocument
APRAKSTS	Izveido jaunu dokumentu autentificētā lietotāja aktuālās e-pakalpojuma sesijas transakcijā. ***Obligāta, veidojot dokumentu (ja dokumenta definīcija nenosaka citādāk), ir tikai 'cmis:name' īpašība.*** ***Pēc noklusējuma, ja ievaddatos nenorāda citu tipu, dokumenti tiek veidoti ar edk:d:customDocument vai, ja to 'cmis:name' beidzas ar .edoc, edk:d:edoc tipu.*** ***Ja kopā ar dokumenta izveidi, saglabā datni, tad tā jānorāda kā pēdējais multipart/form-data formas lauks. Datnes lauka nosaukums - "content".*** ***Datnes nosaukums primāri tiek ņemts no norādītās cmis:contentStreamFileName īpašības, ja tāda nav norādīta, tad no cmis:name.*** ***Gan cmis:name, gan cmis:contentStreamFileName jāatbilst formāta prasībām: tie nevar sākties vai beigties ar ' ' vai saturēt šādus simbolus: '\', ':', '*', '?', '<', '>', '|', '/', '"'.***

METODES IZSAUKŠANA

Adrese

`POST /api/v1/edk/documents`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons.
- `Content-Type` - `multipart/form-data`.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

Nav.

QUERY parametri

Nav

BODY parametri

- `direction` (*string*) - pazīme, kas norāda, vai dokuments ir lietotāja saņemtais (“`in`”) vai izsūtītais dokuments (“`out`”). Noklusētā vērtība: “`in`”.
- `properties` (*json*) - norādāmo īpašību (identifikators (`id`) un vērtību (`value`) saraksts) saraksts, obligāts. Dokumenta īpašību vērtības jānorāda atbilstoši to īpašību tipiem skat. 2. tabulu.
- `additionalAces` (*json*) - papildus pievienojamie pieejas kontroles ierakstu (*ACE*) vārdnīca (atslēga: lietotājs (*principal*), vērtība: pievienojamo tiesību saraksts).
- `content` (*stream*) - dokumenta datne (var būt obligāta, ja to nosaka dokumenta tips).

> ***Šim parametram jābūt beidzamajam formas laukam.***

Piemērs:

```
POST /api/v1/edk/documents HTTP/1.1
Content-Type:     multipart/form-data;     boundary=--------------------------
704361580103470037895673
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377

----------------------------704361580103470037895673
Content-Disposition: form-data; name="direction"

out
----------------------------704361580103470037895673
Content-Disposition: form-data; name="properties"
Content-Type: application/json; charset=utf-8

    [{ "id": "cmis:description",
    "value":[
     "Apraksts1"
    ]
    },
      {
  "id": "cmis:name",
    "value":[
     "Dokuments1.pdf"
    ]
    }
 ]

----------------------------704361580103470037895673
Content-Disposition: form-data; name="additionalAces"
Content-Type: application/json; charset=utf-8

{
 "principal1" : [
   "cmis:all",
   "cmis:read"
 ],
 "principal2" : [
   "cmis:read"
 ]
}
----------------------------704361580103470037895673
Content-Disposition: form-data; name="content"; filename="datnesnosaukums.pdf"
Content-Type: application/octet-stream

dokumenta_datnes_satura_baiti
       ----------------------------704361580103470037895673--
```

IZVADDATI

Izveidotā *EDK* dokumenta identifikators vai kļūda.

Kļūdas

- `HTTP 400` - nav norādīts īpašību saraksts
- `HTTP 400` - norādīts neeksistējošs dokumenta tips.
- `HTTP 400` - norādīta datne, ja dokumenta tips neatbalsta datni.
- `HTTP 400` - nav norādīta datne, ja dokumenta tips nosaka, ka datne ir obligāta
- `HTTP 400` - dokumenta nosaukums satur neatļautos simbols vai neatbilst formātam.

- `HTTP 400` - dokumenta datnes nosaukums satur neatļautos simbols vai neatbilst formātam.
- `HTTP 400` - dokumenta nosaukums nav unikāls e-pakalpojuma transakcijas ietvaros.
- `HTTP 400` - tiek mēģināts norādīt sistēmas noteiktu vai lasīšanas režīmā esošu īpašību (piemēram, `edk:sender`, `edk:receiver`, `edk:retentionGroup`, `edk:owner`, `edk:transaction`)
- `HTTP 400` - norādīta īpašības vērtība, kas neatbilst dokumenta tipa definīcijai, piemēram, pārāk liela vai maza skaitliskā vērtība; norādītas vairākas vērtība, ja tiek nodrošināta tikai viena u. tml.
- `HTTP 400` - nav norādīta īpašības vērtība dokumenta tipa obligāti norādāmajām īpašībām.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=utf-8

URN:IVIS:200266:DOC-1000095734-V1.0
```

7.3.4. Dokumenta īpašību aktualizēšana

IDENTIFIKATORS	LvpContext.EdkService.UpdateDocumentProperties
APRAKSTS	Aktualizē autentificētā lietotāja aktuālās e-pakalpojuma sesijas transakcijā esoša *EDK* dokumenta aprakstošo informāciju (tikai metadatus, t.i., īpašību sarakstu atbilstoši objekta tipa definīcijai). ***Paralēli NEDRĪKST veikt vairākas aktualizēšanas (īpasību vai datnes aktualizēšanu, vai kopīgošanu) manipulācijas ar vienu dokumentu.***

METODES IZSAUKŠANA

Adrese

`PUT /api/v1/edk/documents/{documentId}`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons.
- `Content-Type` - `application/json; charset=utf-8`.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

- `documentId` – aktualizējamā dokumenta identifikators.

QUERY parametri

Nav.

BODY parametri

- `Properties` (*json*) – aktualizējamās īpašības (identifikators (`id`) un vērtību (`value`) saraksts) saraksts, obligāts. Dokumenta īpašību vērtības jānorāda atbilstoši to īpašību tipiem skat. 2. tabulu. Ja ir nepieciešams dzēst kādu vērtību, tad jebkura veida īpašībai kā vērtību jānorāda `JSON NULL`.

Piemērs

```
PUT /api/v1/edk/documents/URN:IVIS:200266:DOC-1000095734-V1.0 HTTP/1.1

Content-Type: application/json; charset=utf-8
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377

{
"properties": [
    {
      "id": "cmis:description",
        "value":[
              "Aktualizēts Apraksts1"
        ]
     }
  ]
}
```

IZVADDATI

`HTTP 204` vai kļūda.

Kļūdas

- `HTTP 401` - lietotājam nav tiesību uzstādīt dokumenta īpašības.
- `HTTP 404` - dokuments neeksistē *EDK* vai tas neietilpst aktuālās transakcijas mapes hierarhijā.
- `HTTP 400` - nav norādīts īpašību saraksts
- `HTTP 400` - tiek mēģināts norādīt sistēmas noteiktu vai lasīšanas režīmā esošu īpašību (piemēram, `edk:sender`, `edk:receiver`, `edk:retentionGroup`, `edk:owner`, `edk:transaction`)
- `HTTP 400` - norādīta īpašības vērtība, kas neatbilst dokumenta tipa definīcijai, piemēram, pārāk liela/maza skaitliskā vērtība; norādītas vairākas vērtība, ja tiek nodrošināta tikai viena.
- `HTTP 400` – nav norādīta īpašības vērtība dokumenta tipa obligāti norādāmajām īpašībām.
- `HTTP 400` - dokumentam ir jaunāka versija.

Piemērs

```
HTTP/1.1 204 No Content
```

7.3.5. Dokumenta datnes izgūšana

IDENTIFIKATORS	LvpContext.EdkService.GetDocumentContent
APRAKSTS	Izgūst gan autentificētā lietotāja aktuālās e-pakalpojuma sesijas transakcijā, gan citur (piemēram, lietotāja iepriekšējo transakcijās) esoša dokumenta datni kā *stream*. Ja tiek izgūta EDK dokumenta, kas atrodas ārpus autentificētā lietotāja aktuālās vai vēsturiskajām transakcijām, datne, piemēram, EDK dokumentu veido kāda cita sistēma, tad šīs sistēmas EDK integrācijas procesā ir nepieciešams nodrošināt to, ka autentificētājam lietotājam attiecīgajam EDK dokumentam ir piešķirta vismaz `edk:getContentStream` tiesība). ***Paralēli NEDRĪKST veikt vairākas aktualizēšanas (īpašību vai datnes aktualizēšanu, vai kopīgošanu) manipulācijas ar vienu dokumentu.***

METODES IZSAUKŠANA

Adrese

`GET /api/v1/edk/documents/{documentId}`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

- `documentId` – dokumenta identifikators, kura datni nepieciešams izgūt.

QUERY parametri

Nav.

BODY parametri

Nav.

Piemērs

```
GET /api/v1/edk/documents/URN:IVIS:200266:DOC-1000095734-V1.0/content HTTP/1.1

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377
```

IZVADDATI

`HTTP 204` vai kļūda.

Kļūdas

- `HTTP 401` - lietotājam nav tiesību izgūt dokumenta datni.
- `HTTP 404` - dokuments neeksistē *EDK* vai tas neietilpst aktuālās transakcijas mapes hierarhijā.
- `HTTP 404` - dokuments eksistē, bet tam nav datnes (serviss informē, ka tieši datne neeksistē).

Piemērs

```
HTTP/1.1 200 OK
Content-Type: application/octet-stream

dokumenta_datnes_satura_baiti
```

7.3.6. Dokumenta datnes aktualizēšana

IDENTIFIKATORS	LvpContext.EdkService.UpdateDocumentContent
APRAKSTS	Pievieno vai aktualizē autentificētā lietotāja aktuālās e-pakalpojuma sesijas transakcijā esoša *EDK* dokumenta datni kā *stream*. ***Ir iespējams mainīt dokumenta datnes nosaukumu (`cmis:contentStreamFileName`), norādot jauno nosaukumu `'filename'` parametra vērtībā.*** ***Datnes nosaukumam jāatbilst formāta prasībām: tas nevar sākties vai beigties ar ' ' vai saturēt šādus simbolus: '\', ':', '*', '?', '<', '>', '|', '/', '"'.*** ***Paralēli NEDRĪKST veikt vairākas aktualizēšanas (īpašību vai datnes aktualizēšanu, vai kopīgošanu) manipulācijas ar vienu dokumentu.***

METODES IZSAUKŠANA

Adrese

`PUT /api/v1/edk/documents/{documentId}/content`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons.
- `Content-Type` – `multipart/form-data`.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

- `documentId` – dokumenta identifikators, kura datni nepieciešams saglabāt.

QUERY parametri

Nav.

BODY parametri

- `filename` (*string*) – datnes nosaukums, neobligāts. Ja netiek norādīts, tad tiek izmantots cmis:name vai iepriekš definētais datnes nosaukums (`cmis:contentStreamFileName`).
- `content` (*stream*) - dokumenta datne, obligāts.

> ***`Content` parametram jābūt beidzamajam formas laukam.***

Piemērs

```
PUT /api/v1/edk/documents/URN:IVIS:200266:DOC-1000095734-V1.0/content HTTP/1.1

Content-Type:     multipart/form-data;     boundary=--------------------------
704361580103470037895673
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377

----------------------------704361580103470037895673
Content-Disposition: form-data; name="filename"

aktualizetais_datnes_nosaukums.pdf
----------------------------704361580103470037895673
Content-Disposition: form-data; name="content"; filename="datnesnosaukums.pdf"
Content-Type: application/octet-stream

aktualizetie_dokumenta_datnes_satura_baiti

----------------------------704361580103470037895673--
```

IZVADDATI

`HTTP 204` vai kļūda.

Kļūdas

- `HTTP 401` - lietotājam nav tiesību aktualizēt datni.
- `HTTP 404` - dokuments neeksistē *EDK* vai tas neietilpst aktuālās transakcijas mapes hierarhijā.
- `HTTP 400` - nav norādīta datne.
- `HTTP 400` - dokumenta tips nosaka, ka datne ir nepārrakstāma (*read only*).
- `HTTP 400` - dokumentam ir jaunāka versija.

Piemērs

```
HTTP/1.1 204 No Content
```

7.3.7. Dokumenta kopīgošana

IDENTIFIKATORS	LvpContext.EdkService.ShareDocument
APRAKSTS	Metodes nolūks ir nodrošināt dokumenta ievietošanu *EDK* mapēs, kas neietilpst autentificētā e-pakalpojuma lietotāja transakcijā. Tiek izmantota EDK uzvedība, kas nodrošina to, ka, ievietojot dokumentu kādā mapē, dokuments manto attiecīgās mapes piekļuves tiesības jeb *ACL*, t.i., izmantojot šo metodi lietotājs var faktiski piešķirt tiesības uz attiecīgo dokumentu citiem *EDK* lietotājiem, piemēram, ārējām sistēmām. Metode pievieno autentificētā lietotāja aktuālās e-pakalpojuma sesijas transakcijā esošu dokumentu norādītajai *EDK* mapei. > ***Lai EDK mapē varētu kopīgot dokumentu, mapei jābūt piešķirtām edk:getProperties un edk:createFolder tiesības EDK autentificētam lietotājam (speciāla EDK lietotāja loma, kas konfigurējas katram repozitorijam individuāli). Šī lietotāja tiesības netiek piešķirtas kopīgotajam dokumentam. Ja šī loma nav definēta vai nav pievienotas norādītās tiesības attiecīgajai EDK mapei (kuras ceļu norāda šīs metodes parametros), tiks izvadīta kļūda. E-pakalpojuma izstrādātajam ir jāpieprasa, lai EDK sistēmas uzturētājs (VRAA) nodrošina attiecīgās tiesības sistēmas mapei, ar kuru e-pakalpojumam nepieciešama dokumentu kopīgošanas funkcionalitāte.*** Faktiski mapē tiek izveidota mapju hierarhija, kas satur lietotāju un transakciju identificējošas mapes (lai nodrošinātu to, ka vienā mapē atrodas tikai vienas transakcijas dokumenti, un e-pakalpojums var kontrolēt dokumentu nosaukumu unikalitāti) un norādi uz dokumentu – pats dokuments kopēts netiek. Norādi no paša objekta nevar atšķirt. Ārējām sistēmā, kuras izmanto šos dokumentus tiek rekomendēts nepaļauties un neiekodēt loģiku, kas balstīta uz izveidoto mapju hierarhiju, kurā atradīsies dokumenta norāde un izmantot *EDK GetContentStream, GetProperties* metodes (lai atrastu pēc EDK dokumenta identifikatora – URN) vai *EDK Query* metodi (lai atrastu pēc unikāla dokumenta nosaukuma). Ar šīm metodēm ir iespējams izgūt šos dokumentus neatkarīgi no tā, kādā mapju hierarhijā tie atrodas. > ***Paralēli NEDRĪKST veikt vairākas aktualizēšanas (īpašību vai datnes aktualizēšanu, vai kopīgošanu) manipulācijas ar vienu dokumentu.***

METODES IZSAUKŠANA

Adrese

`POST /api/v1/edk/documents/{documentId}/folders`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons.
- `Content-Type` - `application/json; charset=utf-8`.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

- `documentId` – dokumenta identifikators, kuru nepieciešams kopīgot.

QUERY parametri

Nav.

BODY parametri

- `path` (*string*) – absolutais mapes ceļš, kurā ievietot dokumenta kopiju.

Piemērs

```
POST          /api/v1/edk/documents/URN:IVIS:200266:DOC-1000095733-V1.0/folders
HTTP/1.1

Content-Type: application/json; charset=utf-8
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377

{
"path": "/CITAS_SISTEMAS_MAPE"
}
```

IZVADDATI

`HTTP 204` vai kļūda.

Kļūdas

- `HTTP 401` - lietotājam nav tiesību ievietot dokumentu norādītajā mapē (nav `cmis:addObjectToFolder` tiesība attiecīgās mapes objektam).
- `HTTP 404` - dokuments neeksistē *EDK* vai tas neietilpst aktuālās transakcijas mapes hierarhijā.
- `HTTP 400` - absolutais ceļš uz mapi nenorāda uz eksistējošu mapi vai lietotājam nav tiesību attiecīgajā hierarhijā veidot mapes (nav `edk:createFolder` tiesība attiecīgās mapes objektam).

Piemērs

```
HTTP/1.1 204 No Content
```

7.3.8. Dokumenta kopīgošanas pārtraukšana

IDENTIFIKATORS	LvpContext.EdkService.StopSharingDocument
APRAKSTS	Izņem autentificētā lietotāja aktuālās e-pakalpojuma sesijas transakcijā esošu dokumentu no *EDK* mapes, kurā tas ir kopīgots. To var darīt tikai tad, ja dokuments atrodas transakcijas mapes hierarhijā. ***Dokumentu kopīgošanu var pārtraukt tikai transakcijas izpildes laikā, kamēr dokuments vēl atrodas lietotāja transakcijas mapju hierarhijā.*** Noņemta tiek norāde uz objektu – pats objekts netiek dzēsts. ***Paralēli NEDRĪKST veikt vairākas aktualizēšanas (īpašību vai datnes aktualizēšanu, vai kopīgošanu) manipulācijas ar vienu dokumentu.***

METODES IZSAUKŠANA

Adrese

`DELETE /api/v1/documents/{documentId}/folders?path={path}`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

- `documentId` – dokumenta identifikators, kuru kopīgošanu nepieciešams pārtraukt.

QUERY parametri

- `path` – absolūtais *EDK* mapes ceļš, no kuras nepieciešams izņemt dokumenta kopiju.

BODY parametri

Nav.

Piemērs

```
DELETE                    /api/v1/edk/documents/URN:IVIS:200266:DOC-1000095733-
V1.0/folders?path=/ CITAS_SISTEMAS_MAPE HTTP/1.1

Content-Type: application/json; charset=utf-8
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377
```

IZVADDATI

`HTTP 204` vai kļūda.

Kļūdas

- `HTTP 401` - lietotājam nav tiesību izņemt dokumentu no norādītās mapes.
- `HTTP 404` - dokuments neeksistē *EDK* vai tas neietilpst aktuālās transakcijas mapes hierarhijā.
- `HTTP 400` - absolūtais ceļš uz mapi nenorāda uz eksistējošu mapi vai lietotājam nav tiesību attiecīgajā hierarhijā veidot mapes (nav `cmis:createFolder` tiesība attiecīgās mapes objektam).

Piemērs

```
HTTP/1.1 204 No Content
```

7.4. LvpContext.Notification

Šajā nodaļā aprakstītas metodes e-pakalpojuma sadarībai ar Notifikāciju servisu

Visas apakšnodaļā aprakstītās metodes ir iespējams izsaukt tikai autentificētā veidā, izmantojot *LVP.IdentityProvider* izsniegtu un izpildes laikā derīgu *OAuth2* (*JWT* vai *References*) talonu.

Visu metožu darbībai nepieciešama aktīva e-pakalpojuma izpildes transakcija, kuru var izveidot, izmantojot *LvpContext.Request* metodi (skat. 7.2.1. sadaļu), attiecīgā lietotāja sesijā.

7.4.1. KDV ziņojuma sūtīšana

IDENTIFIKATORS	LvpContext.NotificationService.SendKdvNotification
APRAKSTS	Inicializē asinhrona ziņojuma nosūtīšanu autentificētājā lietotāja VAI norādītā lietotāja (parametrs `receiverId`, skat. metodes izsaukšana) klienta darba vietai (latvija.lv).

METODES IZSAUKŠANA

Adrese

`POST /api/v1/notification/kdvnotifications`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons.

- `Content-Type` - `application/json; charset=utf-8`.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

Nav.

QUERY parametri

Nav.

BODY parametri

- `title` (*string*) - ziņojuma virsraksts, obligāts.
- `bodyText` (*string*) - ziņojuma teksts, ar ko tiek aizpildīts ziņojuma saturs (*body*). Var saturēt jebkādu tekstu. ziņojuma saturs tiek veidots, šeit norādīto tekstu ievietojot predefinētā *XML* struktūrā. Ja šis parametrs ir norādīts, parametriem `bodyType` un `bodyTransformation` tiek piešķirtas noklusētās vērtības, ja vien tās nav norādītas pieprasījumā.
- `bodyXml` (*string*) - ziņojuma dati, ar ko tiek aizpildīts ziņojuma saturs (*body*). Parametrs tiek izmantots, ja nav norādīts `bodyText` parametrs. Vērtībai jāsatur korekts *XML* simbolu virknes (*string*) formātā.
- `bodyType` (*string*) - ziņojuma datu *VISS XML* resursu katalogā esošas shēmas identifikators (*URN*). Obligāts, ja tiks izmantots `bodyXml` parametrs, t.i., ir norādīts `bodyXml` un nav norādīts `bodyText` parametrs.
- `bodyTransformation` (string) - ziņojuma datu *VISS* resursu katalogā esošas noformēšanas transformācijas (*XSLT*) identifikators (*URN*).
- `discardAfter` (*string*) - datums un laiks, pēc kura jādzēš neizsūtīts (asinhronais) ziņojums to neizsūtot. Vērtībai jābūt "*yyyy-MM-ddTHH:mm:ss.fffZ*", "*yyyy-MM-ddTHH:mm:ssZ*" vai "*yyyy-MM-ddZ*" formātā. Nedrīkst norādīt pagātnes laiku.
- `postponeUntil` (*string*) - datums un laiks, līdz kuram aizturēt ziņojuma sūtīšanu. Vērtībai jābūt "*yyyy-MM-ddTHH:mm:ss.fffZ*", "*yyyy-MM-ddTHH:mm:ssZ*" vai "*yyyy-MM-ddZ*" formātā. Nedrīkst norādīt pagātnes laiku.
- `receiverId` (*string*) - saņēmēja identifikators (*VISS nameidentifier*), kuram sūtīt *KDV* ziņojumu. Pēc noklusējuma tiek sūtīts autentificētām lietotājam.

Piemērs

```
POST /api/v1/notification/kdvnotifications HTTP/1.1

Content-Type: application/json; charset=utf-8
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377

{
"transactionId":"URN:IVIS:100001:EF.VISS-EF00-v1-0-TR-43",
"title":"ziņojums 123",
"receiverId":"PK:11111111111"
}
```

IZVADDATI

Izveidotā asinhronā ziņojuma identifikators (*GUID* simbolu virkne) vai kļūda.

Kļūdas

- `HTTP 401` – norādīts nederīgs autentifikācijas talons.
- `HTTP 400` – nav norādīts obligāts parametrs.
- `HTTP 400` – norādīts *VISS XML* resursu katalogā neeksistējošs ziņojuma datu *XML* shēmas identifikators (bodyType).
- `HTTP 400` – norādīts *VISS* resursu katalogā neeksistējošs ziņojuma datu noformēšanas transformācijas (*XSLT*) identifikators (`bodyTransformation`).

Piemērs

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=utf-8

e23468d558df4dff8c322bede8a91364
```

7.4.2. E-pasta ziņojuma sūtīšana

IDENTIFIKATORS	LvpContext.NotificationService.SendEmailNotification
APRAKSTS	Inicializē asinhrona ziņojuma nosūtīšanu autentificētājā lietotāja profilā VAI norādītā lietotāja (parametrs `receiverId`, skat. metodes izsaukšana) profilā VAI pieprasījumā norādītajai (parametrs `receiverEmail`, skat. metodes izsaukšana) norādītajai e-pasta adresei.

METODES IZSAUKŠANA

Adrese

`POST /api/v1/notification/emailnotifications`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons.
- `Content-Type` - `application/json; charset=utf-8`.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

Nav.

QUERY parametri

Nav.

BODY parametri

- `title` (*string*) - ziņojuma virsraksts, obligāts.
- `bodyText` (*string*) - ziņojuma teksts, ar ko tiek aizpildīts ziņojuma saturs (*body*). Var saturēt jebkādu tekstu. ziņojuma saturs tiek veidots, šeit norādīto tekstu ievietojot predefinētā *XML* struktūrā. Ja šis parametrs ir norādīts, parametriem `bodyType` un `bodyTransformation` tiek piešķirtas noklusētās vērtības, ja vien tās nav norādītas pieprasījumā.
- `bodyXml` (*string*) - ziņojuma dati, ar ko tiek aizpildīts ziņojuma saturs (*body*). Parametrs tiek izmantots, ja nav norādīts `bodyText` parametrs. Vērtībai jāsatur korekts *XML* simbolu virknes (*string*) formātā.
- `bodyType` (*string*) - ziņojuma datu *VISS XML* resursu katalogā esošas shēmas identifikators (*URN*). Obligāts, ja tiks izmantots `bodyXml` parametrs, t.i., ir norādīts `bodyXml` un nav norādīts `bodyText` parametrs.

- `bodyTransformation` (*string*) - ziņojuma datu *VISS* resursu katalogā esošas noformēšanas transformācijas (*XSLT*) identifikators (*URN*).
- `discardAfter` (*string*) - datums un laiks, pēc kura jādzēš neizsūtīts (asinhronais) ziņojums to neizsūtot. Vērtībai jābūt "*yyyy-MM-ddTHH:mm:ss.fffZ*", "*yyyy-MM-ddTHH:mm:ssZ*" vai "*yyyy-MM-ddZ*" formātā. Nedrīkst norādīt pagātnes laiku.
- `postponeUntil` (*string*) - datums un laiks, līdz kuram aizturēt ziņojuma sūtīšanu. Vērtībai jābūt "*yyyy-MM-ddTHH:mm:ss.fffZ*", "*yyyy-MM-ddTHH:mm:ssZ*" vai "*yyyy-MM-ddZ*" formātā. Nedrīkst norādīt pagātnes laiku.
- `receiverId` (*string*) - saņēmēja identifikators (*VISS nameidentifier*), kuram sūtīt e-pasta ziņojumu. Pēc noklusējuma tiek sūtīts autentificētajam lietotājam, uz attiecīgā lietotāja profilā norādīto e-pastu. Vērtība netiek izmantota, ja ir norādīts `receiverEmail` parametrs.
- `receiverEmail` (*string*) - saņēmēja, kuram sūtīt e-pasta ziņojumu, e-pasta adrese. Izmantojot šo parametru, ir iespējams sūtīt ziņojumus e-pasta adresēm, kas nav reģistrētas lietotāju profilos.

Piemērs

```
POST /api/v1/notification/emailnotifications HTTP/1.1

Content-Type: application/json; charset=utf-8
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377

{
  "transactionId":"URN:IVIS:100001:EF.VISS-EF00-v1-0-TR-43",
  "title":"ziņojums 123",
  "receiverId":"PK:11111111111"
}
```

IZVADDATI

Izveidotā asinhronā ziņojuma identifikators (*GUID* simbolu virkne) vai kļūda.

Kļūdas

- `HTTP 401` – norādīts nederīgs autentifikācijas talons.
- `HTTP 400` – nav norādīts obligāts parametrs.
- `HTTP 400` – norādīts *VISS XML* resursu katalogā neeksistējošs ziņojuma datu *XML* shēmas identifikators (bodyType).
- `HTTP 400` - norādīts *VISS* resursu katalogā neeksistējošs ziņojuma datu noformēšanas transformācijas (*XSLT*) identifikators (`bodyTransformation`).

Piemērs

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=utf-8

ff715318c65d48b38ea14bebc040a3d8
```

7.5. **LvpContext.UserProfile**

Šajā nodaļā aprakstītas metodes e-pakalpojuma sadarbībai ar Lietotāja Profila servisu.

Visas apakšnodaļā aprakstītās metodes ir iespējams izsaukt tikai autentificētā veidā, izmantojot *LVP.IdentityProvider* izsniegtu un izpildes laikā derīgu *OAuth2* (*JWT* vai References) talonu.

Profila īpašības var būt pieejamas tikai lasīšanas režīmā (*read only*), īpašību vērtībām var būt pievienota to satura validācija. Šos ierobežojumus definē lietotāja profilu īpašību definīcijās, bet tos nav iespējams priekšlaicīgi noteikt, izmantojot pieejamās metodes. Ja tiks veikta neatļauta darbība, tad tiks izvadīta kļūda.

E-pakalpojumu katalogā definētās īpašību rakstīšanas un lasīšanas tiesības lietotāju grupām (Company un Person) nav iespējams priekšlaicīgi noteikt, izmantojot pieejamās metodes. Ja tiks veikta neatļauta darbība, tad tiks izvadīta kļūda.

3.tabula

Lietotāja profila īpašību vērtību JSON datu tipi

PROFILA ĪPAŠĪBAS TIPS	JSON TIPS	PIEZĪMES
`string`	`JSON String`	
`dateTime`	`JSON String`	Ievaddatos atļautie formāti: "*yyyy-MM-ddTHH:mm:ss.fffZ*". Izvadē tiek izvadīts "*yyyy-MM-ddTHH:mm:ss.fffZ*".
`integer`	`JSON Number`	
`boolean`	`JSON True` vai `JSON False`	
`xml`	`JSON String`	

7.5.1. Īpašību saraksta izgūšana

IDENTIFIKATORS	LvpContext.UserProfileService.GetProperties
APRAKSTS	Izgūst autentificētā lietotāja profila īpašības. ***Lai izpildītu šo metodi, nepieciešama aktīva e-pakalpojuma transakcija attiecīgā lietotāja sesijā.***

METODES IZSAUKŠANA

Adrese

```
GET
/api/v1/userprofile/properties?propertyNames={propertyName1}&propertyNames={propertyName2}
```

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons.
- `Content-Type` - nav jānorāda.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

Nav.

QUERY parametri

- `propertyNames` – izgūstamo īpašību nosaukumi. Ja nenorāda, tad izgūst visas īpašības.

> ***Norādītajiem īpašību nosaukumiem jābūt eksistējošiem attiecīgā lietotāja veida (fiziska pers., juridiska pers., pilnvarotais u. tml.) īpašību definīcijā.***

BODY parametri

Nav.

Piemērs

```
GET /api/v1/userprofile/properties HTTP/1.1

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377
```

IZVADDATI

Lietotāju profila īpašību masīvs. Katra īpašība sastāv no tipa (`type`), nosaukuma (`name`) un vērtības (`value`). Vērtība tiek izvadīta tikai tām īpašībām, kurām tā eksistē. īpašību vērtības tiek izvadītas atbilstoši to tipu formātiem, skat. 3. tabulu.

Kļūdas

- `HTTP 401` – norādīts nederīgs autentifikācijas talons.
- `HTTP 400` – kāda no norādītajiem izgūstamajiem īpašību nosaukumiem neeksistē attiecīgā lietotāja veida (fiziska pers., juridiska pers., pilnvarotais u. tml.) īpašību definīcijā.
- `HTTP 400` – lietotājam nav lasīšanas tiesību kādai no izgūstamajām īpašībām.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

[
  {
    "type": "string",
    "name": "street",
    "value": "Brīvības"
  },
  {
    "type": "xml",
    "name": "propertyxml"
    "value": "<xml><p>This is a XML sample</p></xml>"
  },
  {
    "type": "string",
    "name": "propertyWithoutValue"
  },
  {
    "type": "integer",
    "name": "itemsperpage",
    "value": 10
  },
  {
    "type": "boolean",
    "name": "getinfoonemail",
    "value": true
  },
  {
    "type": "dateTime",
    "name": "date",
    "value": "2019-12-13T16:40:40.000Z"
  }
]
```

7.5.2. Īpašības izgūšana

IDENTIFIKATORS	LvpContext.UserProfileService.GetProperty
APRAKSTS	Izgūst autentificētā lietotāja profila īpašību (tipu un vērtību). ***Lai izpildītu šo metodi, nepieciešama aktīva e-pakalpojuma transakcija attiecīgā lietotāja sesijā.***

METODES IZSAUKŠANA

Adrese

`GET /api/v1/userprofile/properties/{propertyName}`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons.
- `Content-Type` - nav jānorāda.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

- `propertyName` - īpašības nosaukums (nav reģistrjūtīgs), obligāts.

QUERY parametri

Nav.

BODY parametri

Nav.

Piemērs

```
GET /api/v1/userprofile/properties/getinfoonemail HTTP/1.1

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377
```

IZVADDATI

Atrastās īpašības tips (`type`) un vērtība (`value`), ja vērtība eksistē, vai kļūda. Īpašības vērtība tiek izvadīta atbilstoši tās tipa formātam, skat. 3. tabulu.

Kļūdas

- `HTTP 400` - norādīts neeksistējošs īpašības nosaukums.
- `HTTP 401` – norādīts nederīgs autentifikācijas talons.
- `HTTP 400` – lietotājam nav lasīšanas tiesību kādai no izgūstamajām īpašībām.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{"value":true,"type":"boolean"}
```

7.5.3. Īpašību definīciju izgūšana

IDENTIFIKATORS	LvpContext.UserProfileService.GetPropertyDefinitions
APRAKSTS	Izgūst īpašību definīciju sarakstu autentificētā lietotāja veidam (piemēram, iedzīvotājs, juridiska persona). atkarībā no talonā esošā *nameidentifier* vērtības).

METODES IZSAUKŠANA

Adrese

`GET api/v1/userprofile/propertyDefinitions`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons.
- `Content-Type` - nav jānorāda.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

Nav.

QUERY parametri

Nav.

BODY parametri

Nav.

Piemērs

```
GET api/v1/userprofile/propertyDefinitions HTTP/1.1

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377
```

IZVADDATI

Atrasto īpašību definīciju masīvs. Katra īpašības definīcija sastāv no nosaukuma (`propertyName`) un tipa (`type`).

Kļūdas

- `HTTP 401` – norādīts nederīgs autentifikācijas talons.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

[{"type":"string","name":"street"},{"type":"xml","name":"favoriteepakservices"
}]
```

7.5.4. Īpašības vērtības aktualizēšana

IDENTIFIKATORS	LvpContext.UserProfileService.SetProperty
APRAKSTS	Uzstāda autentificētā lietotāja profila īpašības vērtību. ***Lai izpildītu šo metodi, nepieciešama aktīva e-pakalpojuma transakcija attiecīgā lietotāja sesijā.***

METODES IZSAUKŠANA

Adrese

`PUT /api/v1/userprofile/properties/{propertyName}`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons.
- `Content-Type` - `application/json; charset=utf-8`.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

- `propertyName` – īpašības nosaukums (nav reģistrjūtīgs), obligāts.

QUERY parametri

Nav.

BODY parametri

- `value` - uzstādāmās īpašības vērtība. Īpašības vērtība jānorāda atbilstoši tās tipa formātam, skat. 3. tabulu. Lai dzēstu vērtību, visu veidu īpašībām ir iespējams uzstādīt `JSON NULL`.

> ***NB! Dažas īpašības ir pieejamas tikai lasīšanas režīmā (to nav iespējams noteikt, izmantojot pieejamās metodes), ja mēģina mainīt vērtību šādām īpašībām, tad tiks izvadīts `HTTP 400`.***

Piemērs

```
PUT /api/v1/userprofile/properties/getinfoonemail HTTP/1.1

Content-Type: application/json; charset=utf-8
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377

{
  "value":true
}
```

IZVADDATI

`HTTP 204` vai kļūda.

Kļūdas

- `HTTP 401` – ja norādīts nederīgs autentifikācijas talons.
- `HTTP 400` – ja mēģina uzstādīt lasīšanas režīmā esošas īpašības vērtību.
- `HTTP 400` – ja norādīts neeksistējošs īpašības nosaukums.
- `HTTP 400` – ja norādīta nekorekta īpašības vērtība, piemēram, īpašības tips ir *XML*, bet tās vērtība neatbilst *XML* shēmai vai cita veida validācijai.
- `HTTP 400` – lietotājam nav rakstīšanas tiesību aktualizējamai īpašībai.

Piemērs

```
HTTP/1.1 204 No Content
```

7.5.5. Īpašību saraksta aktualizēšana

IDENTIFIKATORS	LvpContext.UserProfileService.SetProperties
APRAKSTS	Uzstāda autentificētā lietotāja profila īpašību vērtības. ***Lai izpildītu šo metodi, nepieciešama aktīva e-pakalpojuma transakcija attiecīgā lietotāja sesijā.***

METODES IZSAUKŠANA

Adrese

`PUT /api/v1/userprofile/properties`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons.
- `Content-Type` - `application/json; charset=utf-8`.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`, `plain/text`.

PATH parametri

Nav.

QUERY parametri

Nav.

BODY parametri

- `properties` - uzstādāmo īpašības saraksts, obligāts. Sarakstam jāsatur vismaz viena uzstādāmā īpašība. Katrai sarakstā esošajai īpašībai ir `name` un `value` atribūti:
  - `name` - īpašības nosaukums (nav reģistrjūtīgs), obligāts;

- `value` - uzstādāmās īpašības vērtība. Īpašību vērtības jānorāda atbilstoši to tipu formātiem, skat. 3. tabulu. Lai dzēstu vērtību, visu veidu īpašībām ir iespējams uzstādīt `JSON NULL`.

> ***NB! Dažas īpašības ir pieejamas tikai lasīšanas režīmā (to nav iespējams noteikt, izmantojot pieejamās metodes), ja mēģina mainīt vērtību šādām īpašībām, tad tiks izvadīts `HTTP 400`.***

Piemērs

```
PUT /api/v1/userprofile/properties HTTP/1.1

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377

{
"properties": [
{
  "name":"position",
  "value": "test"
 }]
}
```

IZVADDATI

`HTTP 204` vai kļūda.

Kļūdas

- `HTTP 401` – norādīts nederīgs autentifikācijas talons.
- `HTTP 400` – mēģina uzstādīt lasīšanas režīmā esošas īpašības vērtību.
- `HTTP 400` – norādīts neeksistējošs īpašības nosaukums.
- `HTTP 400` – norādīta nekorekta īpašības vērtība, piemēram, īpašības tips ir *XML*, bet tās vērtība neatbilst *XML* shēmai vai cita veida validācijai.
- `HTTP 400` – lietotājam nav rakstīšanas tiesību kādai no aktualizējamām īpašībām.

Piemērs

```
HTTP/1.1 204 No Content
```

7.6. LvpContext.Configuration

Šajā nodaļā aprakstītas metodes e-pakalpojuma sadarbībai ar Konfigurācijas servisu.

Visas apakšnodaļā aprakstītās metodes ir iespējams izsaukt bez autentificēšanas.

7.6.1. E-pakalpojuma konfigurācijas izgūšana

IDENTIFIKATORS	LvpContext.ConfigurationApi.GetEserviceConfiguration
APRAKSTS	Izgūst e-pakalpojuma konfigurāciju.

METODES IZSAUKŠANA

Adrese

`GET /api/v1/configuration/eservice/{urn}`

HEADER parametri

- `Content-Type` - nav jānorāda.

- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.

PATH parametri

- urn – e-pakalpojuma identifikators (urlencoded)

QUERY parametri

Nav.

BODY parametri

Nav.

Piemērs

```
GET /api/v1/configuration/eservice/URN%3AIVIS%3A100001%3AEP-EP56-V1-1 HTTP/1.1
```

IZVADDATI

E-pakalpojuma konfigurācijas objekts:

- urn – identifikators;
- type – tips (Lvp, Techincal vai Viss);
- authorityId – iestādes identifikators, kura ir atbildīga par e-pakalpojumu;
- title – nosaukums vairākas valodas;
- url - tīmekļa adrese;
- about – informācija par e-pakalpojumu vairākās valodas;
- instruction – instrukcija vairākas valodas;
- status – status
  - `DRAFT`: Jauna e-pakalpojuma versija, kuras dati vēl tiek ievadīti;
  - `DEVELOPMENT`: Notiek pakalpojuma izstrāde;
  - `PUBLISHED`: Aktīvs;
  - `CLOSED`: Uz laiku slēgta e-pakalpojuma versija;
  - `DELETED`: Dzēsta e-pakalpojuma versija;
- isAnonymous – nosaka, vai e-pakalpojums ir pieejams visiem lietotājiem;
- undeclaredIdentityProviders - nedeklarētās identitātes lietotāju atļauto identitātes sniedzēju saraksts;
- inhabitantIdentityProviders – fiziskās personas lietotāju atļauto identitātes sniedzēju saraksts;
- legalEntityIdentityProviders – juridiskās personas lietotāju atļauto identitātes sniedzēju saraksts;
- scalableInterface – nosaka vai ir pieejams mērogojams interfeiss;
- supportsMultipleLanguages – nosaka vai ir atbalstītas vairākas valodas;
- skipLanguageChangeWarning – nosaka vai izlaist brīdinājumu par valodas maiņu;
- passOnQueryString – nosaka vai pārsūtīt e-pakalpojumam *query string* parametrus, kas saņemti, atverot portāla pakalpojuma lapu;
- cdnVersion - CDN resursu (JavaScript, CSS, attēli) versija, kura jāizmanto portālam, lai nodrošinātu savietojamību ar e-pakalpojuma izmantotajiem resursiem;
- profilePropertiesToRead - lietotāja profila lauki, kurus e-pakalpojums drīkst lasīt;
- profilePropertiesToWrite - lietotāja profila lauki, kurus e-pakalpojums drīkst modificēt;
- price – cenas vērtība vairākas valodas;
- priceDescription - cenas apraksts vairākas valodas;
- receivingTime - saņemšanas laika vērtība vairākas valodas;
- receivingTimeDescription - saņemšanas laika apraksts vairākas valodas;

- receivingResultType - rezultātu saņemšanas vērtība vairākas valodas;
- receivingResultTypeDescription - rezultātu saņemšanas aprakst vairākas valodas;
- restrictions - ierobežojumu vērtība vairākas valodas;
- restrictionsDescription – ierobežojumu aprakst vairākas valodas;
- videoInstructionLink – video instrukcijas saite vairākas valodas;
- videoDescription - video instrukcijas apraksts vairākas valodas;
- instructionSteps – instrukcijas soļi vairākas valodas.

Kļūdas

- `HTTP 204` – norādītājs e-pakalpojuma identifikators nav atrasts.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "urn": "URN:IVIS:100001:EP-EP56-V1-1",
  "type": "Lvp",
  "authorityId": "100564",
  "title": [
    {
      "languageCode": "lv",
      "text": "Informācija par uzturlīdzekļu parādu"
    }
  ],
  "url": "https://app1-dev-vraa.abcsoftware.lv/E-Services/EP56-v1-1",
  "about": [
    {
      "languageCode": "lv",
      "text":    "<p>E-pakalpojuma    &bdquo;Informācija    par    uzturlīdzekļu
parādu</p>"
    }
  ],
  "instruction": [
    {
      "languageCode": "lv",
      "text": ""
    }
  ],
  "status": "Published",
  "isAnonymous": false,
  "undeclaredIdentityProviders": [],
  "inhabitantIdentityProviders": [
    "URN:IVIS:111111:AM-ALLBANKS",
    "URN:IVIS:111111:AM-ALLESIGNATURES"
  ],
  "legalEntityIdentityProviders": [],
  "scalableInterface": true,
  "supportsMultipleLanguages": false,
  "skipLanguageChangeWarning": false,
  "passOnQueryString": false,
  "cdnVersion": "v1.6.2",
  "profilePropertiesToRead": [
    {
      "profileType": "Person",
      "properties": [
        "Email",
        "AddressAtvk",
      ]
    }
  ],
  "profilePropertiesToWrite": [],
  "price": [
          {
                "languageCode": "lv",
                "text": "5"
          },
          {
                "languageCode": "en",
                "text": "Free"
          }
```

```
    ],
    "priceDescription": [
          {
                "languageCode": "lv",
                "text": "Cena"
          },
          {
                "languageCode": "en",
                "text": "Price"
          }
    ],
    "receivingTime": [
          {
                "languageCode": "lv",
                "text": "2min"
          },
          {
                "languageCode": "en",
                "text": "1-2min"
          }
    ],
    "receivingTimeDescription": [
          {
                "languageCode": "lv",
                "text": "Saņemšanas laiks"
          },
          {
                "languageCode": "en",
                "text": "Execution time"
          }
    ],
    "receivingResultType": [
          {
                "languageCode": "lv",
                "text": "Rēķins"
          },
          {
                "languageCode": "en",
                "text": "Bill"
          }
    ],
    "receivingResultTypeDescription": [
          {
                "languageCode": "lv",
                "text": "Izpildes rezultāts"
          },
          {
                "languageCode": "en",
                "text": "Result type"
          }
    ],
    "restrictions": [
          {
                "languageCode": "lv",
                "text": "tikai 18+"
          },
          {
                "languageCode": "en",
                "text": "must be over 18"
```

```
            }
      ],
      "restrictionsDescription": [
            {
                  "languageCode": "lv",
                  "text": "Ierobežojumi"
            },
            {
                  "languageCode": "en",
                  "text": "Restrictions"
            }
      ],
      "videoInstructionLink": [
            {
                  "languageCode": "lv",
                  "text": "https://www.youtube.com/watch?v=pQi3jtNZaIk\"
            },
            {
                  "languageCode": "en",
                  "text": "https://www.youtube.com/watch?v=W8MAT-Fgq04\"
            }
      ],
      "videoDescription": [],
},
"instructionSteps": [
            {
                  "title": [
                        {
                              "languageCode": "lv",
                              "text": "asdas"
                        },
                        {
                              "languageCode": "en",
                              "text": "asds"
                        }
                  ],
                  "description": [
                        {
                              "languageCode": "lv",
                              "text": ""
                        },
                        {
                              "languageCode": "en",
                              "text": ""
                        }
                  ]
            },
            {
                  "title": [
                        {
                              "languageCode": "lv",
                              "text": "Dokumenta izveide"
                        },
                        {
                              "languageCode": "en",
                              "text": "Create document title"
                        }
                  ],
                  "description": [
```

```
                        {
                              "languageCode": "lv",
                              "text":  "Šajā  solī  jums  jāievada  nepieciešamie
dati, lai izveidotu dokumentu"
                        },
                        {
                              "languageCode": "en",
                              "text": "Ener document title"
                        }
                  ]
            },
            {
                  "title": [
                        {
                              "languageCode": "lv",
                              "text": "Step3Test"
                        },
                        {
                              "languageCode": "en",
                              "text": "Step3test"
                        }
                  ],
                  "description": [
                        {
                              "languageCode": "lv",
                              "text": ""
                        },
                        {
                              "languageCode": "en",
                              "text": ""
                        }
                  ]
            },
            {
                  "title": [
                        {
                              "languageCode": "lv",
                              "text": "Sākumums"
                        },
                        {
                              "languageCode": "en",
                              "text": "Intruduction"
                        }
                  ],
                  "description": [
                        {
                              "languageCode": "lv",
                              "text":  "Šajā  solī  jums  tiks  attēlota  vispārēja
informācija par e-pakalpojuma piedāvātajām iespējām"
                        },
                        {
                              "languageCode": "en",
                              "text":  "In  this  step  you  will  see  general
information "
                        }
                  ]
            }
      ]
}
```

7.7. LvpContext.Payments

Detalizētu informāciju par maksājumu API izmantošanu skatīt [7] dokumentā.

7.8. LvpContext.ErrorReport

7.8.1. Kļūdas pieteikuma izveidošana

IDENTIFIKATORS	LvpContext.ErrorApi.ErrorReport
APRAKSTS	Izveido kļūdas pieteikumu.

METODES IZSAUKŠANA

Adrese

`POST api/v1/errorreport/message`

HEADER parametri

- `Content-Type – application/json`
- `X-tabid – 0643ff35bd234e3082d3e994da33c377`

PATH parametri

Nav.

QUERY parametri

Nav.

BODY parametri

Employee – Obligāts. Kopa ar noradītu AuthorityID (iestādes identifikatoru)

EServiceId – Obligāts. E-pakalpojuma identifikators, kurš iesniedz pieteikumu

TransactionId – Obligāts. Esošā lietotāja transakcijas identifikators

Notification – Ziņojuma objekts

Notification.Channel – Kanāls, kurā padod ziņojumu. 0 – tālrunis 1- e-pasts(noklusētais)

Notification.Email – E-pasts uz kuru nosūtīt ziņojumu, ja ir norādīts e-pasta kanāls

Notification.Phone – Tālruņa numurs uz kuru nosūtīt ziņojumu, ja ir norādīts tālruņa kanāls

SystemID – Autentifikācijas piegādātāja URN identifikators

Title – Pieteikuma nosaukums

Message – Pieteikuma teksts

Browser – Klienta izmantotās pārlūkprogrammas informācijas objekts

Browser.Name – Klienta pārlūkprogrammas nosaukums

Browser.Version – Klienta pārlūkprogrammas versija

Browser.UserAgent – Klienta pārlūkprogrammas lietotāja aģents

DisplayResolution – Klienta ekrāna rezolūcija.

Files[] – Masīvs ar pievienotajiem failiem. Pieejams tikai form-data režīmā

KeyValuePairs[] – Masīvs priekš papildus vērtībām konkrētajam gadījumam

KeyValuePairs.Key – Atslēga

KeyValuePairs.Value – Vērtība

IPAddress – Pieteikuma autora klienta IP adrese, noderīga, ja pieprasījumu sūta no servera. Ja nav norādīta, tad tiks paņemta no pieprasījuma galvas.

Piemērs application/json

```
POST /api/v1/errorreport HTTP/1.1

Content-Type: Application/json
x-tabid: 0643ff35bd234e3082d3e994da33c377

{
    "ServiceInstanceID" : "URN:IVIS:100001:EP-IVISErrorReport-v1-1-TR-1709",
    "EserviceId":"URN:IVIS:100001:EP-EP186-v1-0",
    "TransactionId":"URN:IVIS:100001:EP-IVISErrorReport-v1-1-TR-1284",
    "Notification": {
        "Channel": 1,
        "Email":"test.user@gmail.com"
    },
    "SystemID":"URN:IVIS:100003:IDDVApp",
    "Title":"test",
    "Message":"test",
    "Browser": {
        "Name":"Chrome",
        "Version":"83.0",
        "UserAgent":   "Mozilla/5.0   (Windows   NT   10.0;   Win64;   x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/83.0.4103.116 Safari/537.36"
    },
    "DisplayResolution": "1920x1040",
    "KeyValuePairs": [
        {
          "Key":"Url",
          "Value": https://localhost:44321/Home/Service?startANew=true
        },
        {
          "Key":"Step",
          "Value": ""
        }
    ]
}
```

IZVADDATI

Kļūdas

- `HTTP 401` – norādīts nederīgs autentifikācijas talons.
- `HTTP 204` – kļūdas pieteikuma veidošana neizdevās.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
```

7.9. LvpIsolatedContext

Šajā nodaļā aprakstīts e-pakalpojumu infrastruktūras servisu izolācijas / emulācijas serviss, kuru izstrādātājs var izmantot, izstrādājot vai testējot e-pakalpojumu, kad e-pakalpojumu infrastruktūras servisi nav pieejami; nesatur e-pakalpojuma darbībai nepieciešamo konfigurāciju, pielāgojumus vai nenodrošina nepieciešamo API pieejamību.

Serviss nodrošina šādu e-pakalpojumu infrastruktūras servisu funkcionalitātes emulāciju:

- LvpContext.SessionProperties – skat. 7.1. nodaļu. Tiek nodrošināta pilna funkcionalitāte un identiska saskarne. Dati tiek glabāti tikai virtuālajā atmiņā un ir pieejami kamēr dokers ir pacelts.
- LvpContext.Request - skat. 7.1.5. nodaļu. Tiek nodrošināta pilna funkcionalitāte un identiska saskarne. Lai servisu izmantotu, nepieciešams nodrošināt nepieciešamos testa datus (skat. 7.9.2. sadaļu).
- LvpContext.Configuration – skat. 7.6. nodaļu. Tiek nodrošināta pilna funkcionalitāte un identiska saskarne. Lai servisu izmantotu, nepieciešams nodrošināt nepieciešamos testa datus (skat. 7.9.2. sadaļu).
- LvpContext.Access – skat. 7.11. nodaļu. Tiek nodrošināta pilna funkcionalitāte un identiska saskarne. Lai servisu izmantotu, nepieciešams nodrošināt nepieciešamos testa datus (skat. 7.9.2. sadaļu).

7.9.1. Uzstādīšana

Serviss ir pieejams uzstādīšanai kā *Docker* attēls (*Docker image*) izmitināšanai *Docker* konteinerī.

Lai uzstādītu servisu lokālajā izstrādes vidē, e-pakalpojuma izstrādātājs var izmantot *docker compose* rīku, papildinot `docker-compose.yml` ar nepieciešamo konfigurāciju un nodrošinot nepieciešamos vides (*environment*) parametrus (vides parametrus ir iespējams norādīt `.env` datnē, kurai jāatrodas vienā mapē ar `docker-compose.yml`). Daļa no vides parametriem ir nepieciešami paša servisa darbībai (tie ir redzami `docker-compose.yml` konfigurācijas `environmet` sekcijā).

- `docker-compose.yml` konfigurācija:

```
version: '3.8'

services:
  …
  lvp.eserviceplatform.backend.isolatedcontextapi:
    container_name: Lvp.EservicePlatform.Backend.IsolatedContextApi
    environment:
      ISOLATED_CONTEXT_ESERVICE_TRANSACTION_STARTING_NUM:
${ISOLATED_CONTEXT_ESERVICE_TRANSACTION_STARTING_NUM}
      ISOLATED_CONTEXT_DATA_DESTINATION_PATH:
${ISOLATED_CONTEXT_DATA_DESTINATION_PATH}
      LOG_LEVEL: ${LOG_LEVEL}
      ISOLATED_CONTEXT_DISABLE_DEFAULT_SAMPLES:
${ISOLATED_CONTEXT_DISABLE_DEFAULT_SAMPLES}
    # Nexus publiceta Docker image (piemers)
    image: nexus.abc:5001/lvp-eserviceplatform-backend-isolatedcontextapi:1.0.0
    ports:
      - "${ISOLATED_CONTEXT_PORT}:80"
    volumes:
      # Sasaiste starp host datora failu sistemas mapi (SOURCE) un Docker iekejo
failu sistemas mapi (TARGET).
      -
${ISOLATED_CONTEXT_DATA_SOURCE_PATH}:${ISOLATED_CONTEXT_DATA_DESTINATION_PATH}
    restart: on-failure
```

- vides parametri
  - ISOLATED_CONTEXT_DATA_SOURCE_PATH - absolūtais vai relatīvais (pret `docker-compose.yml`) ceļš saknes mapei, kurā eksistē servisa atbilžu izvadīšanai nepieciešamās mapes (`epak_configs`, `rest_responses` un `ivis_responses`) ar `json` datnēm; obligāts. Piemērs: “c:\ep111”.
  - ISOLATED_CONTEXT_DATA_DESTINATION_PATH - absolūtais ceļš *Docker* konteinera iekšienē caur kurieni tiks lasīti ISOLATED_CONTEXT_DATA_SOURCE_PATH norādītājā mapē esošās mapes ar `json` datnēm; obligāts. Piemērs: “/home/app/app_data”.
  - ISOLATED_CONTEXT_PORT – servisa ports, obligāts. Piemērs: “44444”.
  - ISOLATED_CONTEXT_ESERVICE_TRANSACTION_STARTING_NUM – e-pakalpojumu transakciju sākuma numurs; neobligāts. Noklusētā vērtība: “1”.
  - ISOLATED_CONTEXT_DISABLE_DEFAULT_SAMPLES – pazīme noklusēto testa datu atslēgšanai, neobligāts. Noklusētā vērtība: “0”.
  - LOG_LEVEL – žurnāla ierakstu minimālais līmenis (viens no: “Verbose”, “Debug”, “Information”, “Warning”, “Error”, “Fatal”); neobligāts. Noklusētā vērtība: “Information”.

7.9.2. Testa datu konfigurēšana

Tā kā serviss ir izolēts no e-pakalpojumu servisu un aizmugursistēmu infrastruktūras, tad, lai tas varētu veidot transakcijas, izvadīt atbildes integrācijas servisu izsaukumiem, e-pakalpojumu izstrādātājam jānodrošina serviss ar testa datiem.

Serviss testa datus ielasa no ISOLATED_CONTEXT_DATA_SOURCE_PATH mapes, kur tie tiek lasīti no četrām apakšmapēm (citas mapes un datnes tiek ignorētas):

- “eservice_configs“ – mapei jāsatur tikai `utf-8` kodējuma `json` datnes ar e-pakalpojumu konfigurāciju - tieši tādu, kādu izvada LvpContext.Configuration serviss (skat. 7.6. nodaļu). Datnes tiek izmantotas kā e-pakalpojumu katalogs e-pakalpojumu transakciju izveidei (skat. 7.2.1. sadaļu) un LvpContext.Configuration servisa (skat. 7.6. nodaļu) atbilžu izvadīšanai.

- "ivis_responses" – mapei jāsatur tikai `utf-8` kodējuma `json` konfigurācijas datnes ar noteiktu struktūru (skat. 5. tabulu).
- "rest_responses" – mapei jāsatur tikai `utf-8` kodējuma `json` konfigurācijas datnes ar noteiktu struktūru (skat. 5. tabulu).
- "eservices_access" - mapei jāsatur tikai `utf-8` kodējuma `json` konfigurācijas datnes ar noteiktu struktūru (skat. 6. tabulu).

ISOLATED_CONTEXT_DATA_SOURCE_PATH var saturēt tikai tās apakšmapes un testa datus, kas nepieciešami e-pakalpojuma izpildei, t.i., visas apakšmapes nav obligātas. Dati no datnēm tiek ielasīti katra pieprasījuma sākumā un tie netiek kešoti. Ja ielādējot testa datnes ir notikusi kļūda (piemēra, serializācijā), tā paradīsies žurnālā, kas tiek rakstīts konsolē.

Apakšmapēs esošo `json` datņu nosaukumiem nav noteiktu prasību.

4.tabula

API izsaukšanas JSON konfigurācijas datnes struktūras apraksts

LAUKS	OBLIGĀTS	VĒRTĪBA
`TargetUrl`	Jā	Relatīvā API Pārvaldniekā reģistrētā integrācijas servisa adrese, t.i., jābūt vienādai ar simbolu virkni, kura tiks norādīta REST vai IVIS integrācijas servisa izsaukuma "`targetUrl`" parametra vērtībā.
`Body`	Nē	IVIS integrācijas servisu gadījumā jāsatur nepieciešamā IVISResponse struktūra. REST integrācijas servisu gadījumā jāsatur REST servisa atbilde.
`Headers`	Nē	HTTP atbildes galvenes elementu vārdnīca (vērtības jānorāda masīvā). Pēc noklusējuma IVIS integrācijas servisu gadījumā `Content-Type` ir "`application/xml;charset=utf-8`", REST – "`application/json;charset=utf-8`".
`StatusCode`	Nē	Nepieciešamais atbildes HTTP statusa kods. Noklusētā vērtība – 200.
`BaseUrl`	Nē	Absolūtā API Pārvaldnieka bāzes adrese, t.i., reālā integrācijas vai `mock` servisa bāzes adrese (apvienojumā ar `TargetUrl` jāveido pilnā adrese). Ja norādīta, tad atbildē tiek izvadīta reālā servisa atbilde, t.i., `Body`, `Headers` un `StatusCode` vērtības tiek ignorētas. Šo parametru var izmantot, kad e-pakalpojumu izstrādātājam ir pieejams IVIS vai REST integrācijas serviss vai arī IVIS vai REST integrācijas serviss atbildē izvada komplicētu atbildi – multipart formu, stream vai tml.

API izsaukšanai paredzēto JSON konfigurācijas datņu piemēri:

- Konfigurācijas datnes piemērs predefinētas IVIS integrācijas servisa atbildes saņemšanai

```
{
  "TargetUrl": "sample2",
  "Body":                                                    "<IVISResponse
xmlns='http://ivis.eps.gov.lv/XMLSchemas/100001/IVIS/v1-
0'><Header><MessageID>714c97f3-9278-43f4-afbb-
d2c9037d2fd1</MessageID><MessageType>URN:IVIS:100001:XSD-Testing-
```

```
TestISServise-v1-0-TYPE-Result</MessageType><TransactionID>URN:IVIS:100001:EP-EP01-V1-1-TR-1</TransactionID><CorrelationID>77ec1cd6-4bfe-46ee-bc2a-8f629b5325a6</CorrelationID><TimeStamp>2020-07-21T13:19:35.7469032+03:00</TimeStamp><Result>success</Result></Header><Body><Result xmlns='http://ivis.eps.gov.lv/XMLSchemas/100000/TestISServise/v1-0' xmlns:ivis='http://ivis.eps.gov.lv/XMLSchemas/100001/IVIS/v1-0' xmlns:pers='http://ivis.eps.gov.lv/XMLSchemas/100001/Person/v1-0'><CalculationValue>30</CalculationValue></Result></Body></IVISResponse>"
}
```

- Konfigurācijas datnes piemērs predefinētas REST servisa atbildes saņemšanai

```
{
"TargetUrl": "sample1",
"Body": "7.5",
  "Headers": {
    "Content-Type": [
      "application/json;charset=utf-8"
    ],
    "x-test-1": [
      "value1",
      "value2"
    ],
    "x-test-2": [
      "value3"
    ]
  },
  "StatusCode": "201",
  "BaseUrl": null
}
```

- Konfigurācijas datnes piemērs izstrādes stadijā esoša vai izstrādes nolūkiem (mock) paredzēta REST servisa izsaukšanai.

```
{
  "TargetUrl": "sample2",
  "BaseUrl": "https://stackoverflow.com/questions"
}
```

5.tabula

E-pakalojuma piekļuves tiesību pārbaudes JSON konfigurācijas datnes struktūras apraksts

LAUKS	OBLIGĀTS		VĒRTĪBA
Token	Jā		Lietotāja talons. Var būt jebkāda simbolu virkne. Lai konfigurācija tiktu izmantota, vērtībai jāsakrīt ar pieprasījumā norādīto talonu.
Urn	Jā		E-pakalpojuma URN. Lai konfigurācija tiktu izmantota, vērtībai jāsakrīt ar pieprasījumā norādīto vērtību.
UserType	Nē		Lietotāja veids. Emulācijas servisa gadījumā lietotāja veids nevar tikt noteiks, izmantojot talonu, tāpēc tas ir jānorāda konfigurācijā. Pieejamās vērtības:

LAUKS	OBLIGĀTS		VĒRTĪBA
			• 1 – fiziska persona (noklusētā vērtība); • 2 – juridiska persona; • 3 – fiziskas personas pilnvarotā persona; • 4 – juridiskas personas pilnvarotā persona; • 5 – iestādes pilnvarotā persona. ***Lietotāja veids ietekmē servisa uzvedību.***

IsolatedContext satur iebūvētus piemērus:

- IVIS un REST servisiem, kurus ir iespējams izsaukt, norādot `targetUrl` “sample1” vai “sample2” (detalizētāk skat. 7.9.3. sadaļu);
- konfigurāciju diviem e-pakalpojumiem - "`URN:IVIS:100001:EP-EP00-V1-0`" un "`URN:IVIS:100001:EP-EP01-V1-1`";
- Piekļuves tiesību pārbaudei e-pakalpojumam “URN:IVIS:100001:EP-EP00-V1-0” lietotājiem, kuru taloni ir “sample1” (fiziska persona) un “sample2” (juridiskas personas pilnvarotā persona).

Iebūvētos piemērus ir iespējams atslēgt, norādot ISOLATED_CONTEXT_DISABLE_DEFAULT_SAMPLES vides parametrā vērtību “`1`” vai “`true`”.

7.9.3. Iebūvēto API piemēru izsaukšana

Izolācijas / emulācijas serviss satur četrus iebūvētos API – divus REST un divus IVIS API servisiem. Šajā sadaļā aprakstīti precīzi soļi, kurus izpildot, ir iespējams izsaukt šos piemērus.

1. Iebūvētie piemēri nedrīkst būt atslēgti (skat. 7.9.2. sadaļu);
   Lai izsauktu jebkuru API servisu, nepieciešama aktīva transakcija un jābūt norādītam Authorization galvenes elementam (emulācijas servisa gadījumā – Authorization var būt jebkāda simbolu virkne ar vismaz vienu elementu). Izolācijas / emulācijas serviss ļauj izveidot transakcijas tikai tiem e-pakalpojumiem, kuri ir definēti (skat. 7.9.2. sadaļu). Iebūvētajos datos ietilpst 2 e-pakalpojumi (skat. 7.9.2. sadaļu). Transakcija jāveido tieši tāpat, kā 7.2.1. sadaļā kā eServiceId, norādot eksistējošu e-pakalpojumu identifikatoru, piemēram, "`URN:IVIS:100001:EP-EP01-V1-1`". Pieprasījuma piemērs:

```
POST                                                    https://eservices-dev-
vraa.abcsoftware.lv/EservicePlatform.IsolatedContextApi/api/v1/request/transac
tions HTTP/1.1
Content-Type: application/json
x-tabId: 29A752A065DB4C2686C186C8CBF83031
Authorization: Bearer lvp_identityserver_access_token

{
"eServiceId": "URN:IVIS:100001:EP-EP01-V1-1"
}
```

   Atbildes piemērs:

```
HTTP/1.1 200 OK
content-type: text/plain; charset=utf-8

URN:IVIS:100001:EP-EP01-V1-1-TR-32
```

2. Kad ir izveidota transakcija, tad ir iespējams izsaukt:
   - REST API piemērus tieši tāpat kā 7.2.4. sadaļā
     - sample1 – REST servisa piemērs, kur ir predefinēta atbilde.

```
{
  "TargetUrl": "sample1",
  "Body": "7.5",
  "Headers": {
    "Content-Type": [
      "application/json;charset=utf-8"
    ]
  },
  "StatusCode": "200",
  "BaseUrl": null
}
```

Lai izsauktu kā

- `targetUrl` jānorāda "sample1";
- `x-milestoneId` jānorāda "`URN:IVIS:100001:EP-EP01-V1-1-MS-CallCalcSync`" (atkarīgs no izmantota e-pakalpojuma identifikatora);

Pieprasījuma piemērs:

```
GET https://eservices-dev-vraa.abcsoftware.lv/EservicePlatform.IsolatedContextApi/api/v1/request/apirequests?TargetUrl=sample1
x-tabId: 29A752A065DB4C2686C186C8CBF83031
x-milestoneId: URN:IVIS:100001:EP-EP01-V1-1-MS-CallCalcSync
Authorization: Bearer lvp_identityserver_access_token
```

Atbildes piemērs:

```
HTTP/1.1 200 OK
content-type: application/json;charset=utf-8

7.5
```

- sample2 – REST servisa piemērs, kur tiek izmantots reāls resurss (jābūt pieejamam resursam, kas noradīts `BaseUrl` adresē)

```
{
  "TargetUrl": "sample2",
  "Body": null,
  "Headers": null,
  "StatusCode": "200",
  "BaseUrl": "https://stackoverflow.com/questions"
}
```

Lai izsauktu, kā

- `targetUrl` jānorāda "sample2";
- `x-milestoneId` jānorāda "`URN:IVIS:100001:EP-EP01-V1-1-MS-CallCalcSync`" (atkarīgs no izmantota e-pakalpojuma identifikatora);

Pieprasījuma piemērs:

```
GET https://eservices-dev-vraa.abcsoftware.lv/EservicePlatform.IsolatedContextApi/api/v1/request/apirequests?TargetUrl=sample2
x-tabId: 29A752A065DB4C2686C186C8CBF83031
x-milestoneId: URN:IVIS:100001:EP-EP01-V1-1-MS-CallCalcSync
Authorization: Bearer lvp_identityserver_access_token
```

Atbilde:

Tieši tas pats, ko izvadītu https://stackoverflow.com/questions/sample2 izsaukums.

- IVIS API piemērus tieši tāpat kā 7.2.3. sadaļā
  - sample1 – IVIS integrācijas serviss ar predefinētu atbildi ar vietturiem (*placeholders*).

```
{
  "TargetUrl": "sample1",
  "Body": "<IVISResponse xmlns='http://ivis.eps.gov.lv/XMLSchemas/100001/IVIS/v1-0'><Header><MessageID>[[MessageID]]</MessageID><MessageType>[[MessageType]]</MessageType><TransactionID>[[TransactionID]]</TransactionID><CorrelationID>[[CorrelationID]]</CorrelationID><TimeStamp>[[TimeStamp]]</TimeStamp><Result>success</Result></Header><Body><Result xmlns='http://ivis.eps.gov.lv/XMLSchemas/100000/TestISServise/v1-0' xmlns:ivis='http://ivis.eps.gov.lv/XMLSchemas/100001/IVIS/v1-0' xmlns:pers='http://ivis.eps.gov.lv/XMLSchemas/100001/Person/v1-0'><CalculationValue>30</CalculationValue></Result></Body></IVISResponse>",
  "Headers": null,
  "StatusCode": null,
  "BaseUrl": null
}
```

Lai izsauktu, kā

- `targetUrl` jānorāda “sample1”;
- `x-milestoneId` jānorāda “`URN:IVIS:100001:EP-EP01-V1-1-MS-CallCalcSync`” (atkarīgs no izmantota e-pakalpojuma identifikatora);
- `messageType` jānorāda korekti formatēta ziņojuma tipa vērtība, bet tās saturs neietekmē rezultātu;
- `body` jānorāda jebkāds korekts XML – tā saturs netiek ņemts vērā.

Pieprasījuma piemērs:

```
POST https://eservices-dev-vraa.abcsoftware.lv/EservicePlatform.IsolatedContextApi/api/v1/request/ivisrequests?TargetUrl=sample1&messageType=URN:IVIS:100001:XSD-Testing-TestISServise-V1-0-TYPE-Calculation
Content-Type: application/xml; charset=utf-8
x-tabId: 29A752A065DB4C2686C186C8CBF83031
x-milestoneId: URN:IVIS:100001:EP-EP01-V1-1-MS-CallCalcSync
Authorization: Bearer lvp_identityserver_access_token

<test/>
```

Atbildes piemērs:

> ***Šī piemēra gadījumā, jo Body satur vietturus (placeholders), IVISResponse galvenē atribūtu vērtības tiek aizstātas ar aktuālajām vērtībām***

```
HTTP/1.1 200 OK
date: Thu, 24 Sep 2020 09:14:25 GMT
content-type: application/xml;charset=utf-8

<IVISResponse xmlns='http://ivis.eps.gov.lv/XMLSchemas/100001/IVIS/v1-0'>
  <Header>
    <MessageID>4646aec0-232c-4c63-8bc4-5101f04b2e66</MessageID>
    <MessageType>URN:IVIS:100001:XSD-Testing-TestISServise-v1-0-TYPE-
Result</MessageType>
    <TransactionID>URN:IVIS:100001:EP-EP01-V1-1-TR-32</TransactionID>
    <CorrelationID>dd119279-f57b-4588-983c-a395f6098a91</CorrelationID>
    <TimeStamp>2020-09-24T09:14:25.5994065+00:00</TimeStamp>
    <Result>success</Result>
  </Header>
  <Body>
    <Result xmlns='http://ivis.eps.gov.lv/XMLSchemas/100000/TestISServise/v1-0'
            xmlns:ivis='http://ivis.eps.gov.lv/XMLSchemas/100001/IVIS/v1-0'
           xmlns:pers='http://ivis.eps.gov.lv/XMLSchemas/100001/Person/v1-0'>
      <CalculationValue>30</CalculationValue>
    </Result>
  </Body>
</IVISResponse>
```

- sample2 - IVIS integrācijas serviss ar predefinētu atbildi.

```
{
  "TargetUrl": "sample2",
  "Body":                                                    "<IVISResponse
xmlns='http://ivis.eps.gov.lv/XMLSchemas/100001/IVIS/v1-
0'><Header><MessageID>714c97f3-9278-43f4-afbb-
d2c9037d2fd1</MessageID><MessageType>URN:IVIS:100001:XSD-Testing-
TestISServise-v1-0-TYPE-Result</MessageType><TransactionID>URN:IVIS:100001:EP-
EP01-V1-1-TR-1</TransactionID><CorrelationID>77ec1cd6-4bfe-46ee-bc2a-
8f629b5325a6</CorrelationID><TimeStamp>2020-07-
21T13:19:35.7469032+03:00</TimeStamp><Result>success</Result></Header><Body><R
esult      xmlns='http://ivis.eps.gov.lv/XMLSchemas/100000/TestISServise/v1-0'
xmlns:ivis='http://ivis.eps.gov.lv/XMLSchemas/100001/IVIS/v1-0'
xmlns:pers='http://ivis.eps.gov.lv/XMLSchemas/100001/Person/v1-
0'><CalculationValue>30</CalculationValue></Result></Body></IVISResponse>",
  "Headers": null,
  "StatusCode": null,
  "BaseUrl": null
}
```

Lai izsauktu, kā

- `targetUrl` jānorāda “sample2”;
- `x-milestoneId` jānorāda “`URN:IVIS:100001:EP-EP01-V1-1-MS-CallCalcSync`” (atkarīgs no izmantota e-pakalpojuma identifikatora);
- `messageType` jānorāda korekti formatēta ziņojuma tipa vērtība, bet tās saturs neietekmē rezultātu;
- `body` jānorāda jebkāds korekts XML – tā saturs netiek ņemts vērā.

Pieprasījums:

```
POST                                                    https://eservices-dev-
vraa.abcsoftware.lv/EservicePlatform.IsolatedContextApi/api/v1/request/ivisreques
```

```
ts?TargetUrl=sample2&messageType=URN:IVIS:100001:XSD-Testing-TestISServise-V1-0-TYPE-Calculation
Content-Type: application/xml; charset=utf-8
x-tabId: 29A752A065DB4C2686C186C8CBF83031
x-milestoneId: URN:IVIS:100001:EP-EP01-V1-1-MS-CallCalcSync
Authorization: Bearer lvp_identityserver_access_token

<test/>
```

Atbilde:

> ***Šī piemēra gadījumā, jo Body nesatur vietturus (placeholders), IVISResponse galvenē atribūtu vērtības netiek aizstātas ar aktuālajām vērtībām***

```
HTTP/1.1 200 OK
content-type: application/xml;charset=utf-8

<IVISResponse xmlns='http://ivis.eps.gov.lv/XMLSchemas/100001/IVIS/v1-0'>
  <Header>
    <MessageID>714c97f3-9278-43f4-afbb-d2c9037d2fd1</MessageID>
    <MessageType>URN:IVIS:100001:XSD-Testing-TestISServise-v1-0-TYPE-
Result</MessageType>
    <TransactionID>URN:IVIS:100001:EP-EP01-V1-1-TR-1</TransactionID>
    <CorrelationID>77ec1cd6-4bfe-46ee-bc2a-8f629b5325a6</CorrelationID>
    <TimeStamp>2020-07-21T13:19:35.7469032+03:00</TimeStamp>
    <Result>success</Result>
  </Header>
  <Body>
    <Result xmlns='http://ivis.eps.gov.lv/XMLSchemas/100000/TestISServise/v1-0'
            xmlns:ivis='http://ivis.eps.gov.lv/XMLSchemas/100001/IVIS/v1-0'
            xmlns:pers='http://ivis.eps.gov.lv/XMLSchemas/100001/Person/v1-0'>
      <CalculationValue>30</CalculationValue>
    </Result>
  </Body>
</IVISResponse>
```

3. E-pakalpojuma piekļuves tiesību pārbaudes piemēru izpildei nav nepieciešama transakcija. Tos ir iespējams izsaukt tieši tāpat kā 7.11. sadaļā. Emulācijas servisā esošie piemēri:
   - sample1 – piemērs, kur lietotājs ir fiziska persona.

```
{
  "Token": "sample1",
  "Urn": "URN:IVIS:100001:EP-EP00-V1-0",
  "UserType": 1
}
```

Lai izsauktu kā

- `urn` jānorāda `"URN:IVIS:100001:EP-EP00-V1-0"`;
- `Authorization` jānorāda “`Bearer sample1`”;

Pieprasījuma piemērs:

```
GET https://eservices-dev-vraa.abcsoftware.lv/EservicePlatform.IsolatedContextApi/api/v1/access/eservices/URN:IVIS:100001:EP-EP00-V1-0
x-tabId: 29A752A065DB4C2686C186C8CBF83031
Authorization: Bearer sample1
```

Atbildes piemērs:

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{"title":"Viens vai vairaki ievadparametri nav noraditi vai noraditi kludaini.","status":400,"detail":"E-pakalpojuma izpildes pilnvaru parbaude ir paredzeta tikai pilnvarotajam personam."}
```

- sample2 – piemērs, kur, kur lietotājs ir juridiskas personas pilnvarotā persona.

```
{
  "Token": "sample2",
  "Urn": "URN:IVIS:100001:EP-EP00-V1-0",
  "UserType": 4
}
```

Lai izsauktu kā

- `urn` jānorāda `"URN:IVIS:100001:EP-EP00-V1-0"`;
- `Authorization` jānorāda “`Bearer sample2`”;

Pieprasījuma piemērs:

```
GET https://eservices-dev-vraa.abcsoftware.lv/EservicePlatform.IsolatedContextApi/api/v1/access/eservices/URN:IVIS:100001:EP-EP00-V1-0
x-tabId: 29A752A065DB4C2686C186C8CBF83031
Authorization: Bearer sample2
```

Atbildes piemērs:

```
HTTP/1.1 200 OK
content-type: application/json;charset=utf-8

true
```

7.10.LvpContext.Navigation

Šajā nodaļā aprakstītas metodes e-pakalpojuma sadarbībai ar Navigācijas servisu.

Visas apakšnodaļā aprakstītās metodes ir iespējams izsaukt bez autentificēšanas.

7.10.1.Galvenē attēlojamā satura izgūšana

IDENTIFIKATORS	NavigationApi.GetHeader
APRAKSTS	Izgūst galvenē attēlojamos datus.

METODES IZSAUKŠANA

Adrese

`GET EservicePlatform.NavigationApi/api/header`

HEADER parametri

- `Content-Type` - nav jānorāda.

PATH parametri

Nav.

QUERY parametri

- language – neobligāts valodas identifikators. Iespējamās vērtības “lv”, “ne”, “ru”. Ja nenorāda vērtību tiek atgriezts saturs visās valodās.

BODY parametri

Nav.

Piemērs

```
GET /EservicePlatform.NavigationApi/api/header?language=lv HTTP/1.1
```

IZVADDATI

E-pakalpojuma galvenes saturu aprakstošs objekts.

Kļūdas

- `HTTP 400` – satur norādītajā valodā nav atrasts.

Piemērs

```
{
    "headers": {
        "LV": {
            "mastHeadBar": {
                "accessibilityControl": {
                    "accessibilityControlItems": [
                        {
                            "tooltip": "test tooltip",
                            "type": "text"
                        },
                        {
                            "tooltip": "Some tooltip",
                            "type": "theme"
                        }
                    ]
                },
                "languageControl": {
                    "languages": [
                        {
                            "title": "EN",
                            "link": "#"
                        },
                        {
                            "title": "RU”,
                            "link": "#"
                        }
                    ]
                },
                "loginControl": {
                    "link": "#",
                    "title": "Ienākt Mana Latvija.lv"
                },
                "searchControl": {
                    "placeholder": "Meklēt portālā...",
                    "title": "Top meklētākie pakalpojumi",
                    "allResultsTitle": "Visi rezultāti",
                    "searchResultItemGroups": [
                        {
                            "items": [
                                {
                                    "descr": "Some descr",
                                    "header": "Some header text",
                                    "target": "#"
                                },
                                {
                                    "descr": "Some descr",
                                    "header": "Some header text",
                                    "target": "#"
                                }
                            ]
                        },
                        {
                            "items": [
                                {
                                    "descr": "Some descr",
                                    "header": "Some header text",
                                    "target": "#"
                                }
                            ]
```

```
                        }
                    ]
                }
            },
            "navigationBar": {
                "items": [
                    {
                        "title": "E - pakalpojumi",
                        "link": "/epak",
                        "allResultsTitle": "Apskatīt visus e-pakalpojumus",
                        "navigationBarItemCategories": [
                            {
                                "title": "Veselība",
                                "navigationBarItemCategoryItems": [
                                    {
                                        "title": "Iesniegums slimības pabalsta
piešķiršanai (B lapa)",
                                        "link": "#"
                                    },
                                    {
                                        "title": "Darbnespējas lapas",
                                        "link": "#"
                                    }
                                ]
                            }
                        ]
                    }
                ]
            },
            "notificationBar": {
                "alerts": [
                    {
                        "message": "Some alert text",
                        "icon": "ic-warn-round"
                    }
                ]
            }
        }
    }
}
```


7.10.2.Kājenē attēlojamā satura izgūšana

IDENTIFIKATORS	NavigationApi.GetFooter
APRAKSTS	Izgūst kājenē attēlojamos datus.

METODES IZSAUKŠANA

Adrese

`GET EservicePlatform.NavigationApi/api/footer`

HEADER parametri

- `Content-Type` - nav jānorāda.

PATH parametri

Nav.

QUERY parametri

- language – neobligāts valodas identifikators. Iespējamās vērtības “lv”, “ne”, “ru”. Ja nenorāda vērtību tiek atgriezts saturs visās valodās.

BODY parametri

Nav.

Piemērs

```
GET /EservicePlatform.NavigationApi/api/footer?language=lv HTTP/1.1
```

IZVADDATI

E-pakalpojuma kājeni saturu aprakstošs objekts.

Kļūdas

- `HTTP 400` – satur norādītajā valodā nav atrasts.

Piemērs

```
{
    "footers": {
        "LV": {
            "menu": {
                "menuItemGroups": [
                    {
                        "label": "Lapas karte",
                        "items": [
                            {
                                "label": "Apskatīt lapas karti",
                                "target": "#"
                            }
                        ]
                    },
                    {
                        "label": "Noderīgi",
                        "items": [
                            {
                                "label": "Kontakti un saziņa",
                                "target": "#"
                            },
                            {
                                "label": "Par portālu",
                                "target": "#"
                            }
                        ]
                    }
                ]
            },
            "logos": [
                {
                    "alt": "Eraf",
                    "src": "/images/logos/eraf@2x.png",
                    "target": "#"
                },
                {
                    "alt": "eugo",
                    "src": "/images/logos/eugo-logo-rgb-60@2x.png",
                    "target": "#"
                }
            ],
            "contacts": {
                "authorityName": "Valsts reģionālās attīstības aģentūra",
                "address": "Rīga, Alberta iela 10, LV-1010",
                "phone": "tālr.: 67502757",
                "email": "portals@vraa.gov.lv"
            },
            "socials": [
                {
                    "icon": "youtube",
                    "link": "#"
                },
                {
                    "icon": "facebook",
                    "link": "#"
                }
            ]
        }
    }
```

```
}
```

7.11.LvpContext.Access

Šajā nodaļā aprakstītas metodes e-pakalpojuma sadarbībai ar piekļuves pārbaudes servisiem. Visas sadaļā aprakstītās metodes ir iespējams izsaukt tikai autentificētā veidā, izmantojot *LVP.IdentityProvider* izsniegtu un izpildes laikā derīgu *OAuth2* (*JWT* vai *references*) talonu.

7.11.1.E-pakalpojuma izpildes tiesību pazīmes izgūšana

IDENTIFIKATORS	LvpContext.Access.Eservices
APRAKSTS	Izgūst aktuālā lietotāja konkrētā e-pakalpojuma izpildes tiesību (derīgas pilnvaras esamības) pazīmi. Metodi ir paredzēts izmantot tikai pilnvaroto personu gadījumos (fizisku un juridisku personu gadījumā tiek izvadīta kļūda).

METODES IZSAUKŠANA

Adrese

`GET EservicePlatform.ContextAPI/api/v1/access/eservices/{urn}`

HEADER parametri

- `Authorization` - *Bearer OAuth2* (*JWT* vai *references*) talons, obligāts.
- `x-tabId` – aktuālās pārlūka cilnes identifikators, obligāts. Vērtībai jābūt *GUID* formātā.
- `Accept` – vēlamais atbildes formāts, serviss nodrošina `application/json`.

PATH parametri

- `urn` – e-pakalpojuma identifikators (*urlencoded*), obligāts.

QUERY parametri

Nav.

BODY parametri

Nav.

Piemērs

```
GET    EservicePlatform.ContextApi/api/v1/access/eservices/URN:IVIS:100001:EP-EP56-V1-1 HTTP/1.1

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6InNEWX..
x-tabId: 0643ff35bd234e3082d3e994da33c377
```

IZVADDATI

E-pakalpojuma izpildes tiesību esamības gadījumā tiek izvadīts `JSON TRUE`, neesamības – `JSON FALSE`.

Kļūdas

- `HTTP 401` – nenorādīts vai nederīgs autentifikācijas talons.
- `HTTP 400` – nekorekti pieprasījuma dati, piemēram, norādīts fiziskās vai juridiskās personas talons.

Piemērs

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

true
```

8. E-pakalpojumu konteineru izstrāde un piegāde

Detalizētu informāciju par e-pakalpojuma piegādes kārtību skatīt - https://viss.gov.lv/lv/Informacijai/Dokumentacija/Vadlinijas/Piegazu_kartiba.

Detalizētu informāciju par automatizēto piegāžu nodošanas procesu skatīt dokumentā [10].

Lai piegādātu e-pakalpojumu ir nepieciešams:

1. Piegādāt e-pakalpojuma docker image uz VRAA nexus - dokeru repozitoriju;
2. Piegādāt helm skriptus uz VRAA nexus - helm skriptu repozitoriju;
3. Veikt jenkins API izsaukumu un nodot tam informāciju par jaunu piegādi;
4. Jāpiegādā e-pakalpojuma pirmkods uz VRAA Git repozitoriju – VRAA izveidos repozitoriju konkrētajam e-pakalpojumam;
5. Jāsagatavo e-paklpojuma apraksta datne Excel formātā - "E-pakalpojuma apraksta šablons", pieejmas https://viss.gov.lv/lv/Informacijai/Dokumentacija/Koplietosanas_komponentes/EPAK_izstrades_izpildes_vide.
6. Jāsagatavo e-pakalpojuma konfigurācijas yaml datnes piemērs un ja nepieciešams adminstratora rokasgrāmata;
7. Jāsagatavo piegādes readme, ja netiek piegādāta administratora rokasgrāmata jāiekļaut visu konfigurācijas parametru aprakstu;
8. Jāpiegādā VRAA readme, konfigurācijas yaml un apraksta datne.

8.1. E-pakalpojumu konteineru veidošana

Ietvars nodrošina iespēju veidot MPA (multi page application) e-pakalpojumus izmantojot ASP.NET Core MVC ietvaru un SPA (single page application) e-pakalpojumus izmantojot React SDK. E-pakalpojuma konteineru skaits ir atkarīgs no izmantotās izstrādes tehnoloģijas. Parasti, ASP.NET Core MVC e-pakalpojums sastāv no viena konteinera, bet React no diviem konteineriem:

- Frontend: lietojuma prezentācija, kas ir balstīta uz React SDK.
- Backend for Frontend (BFF): darbības loģika, kas ir eksponēta kā API REST saskarne un tiek veidota izmantojot servera darbināto kodu, piemēram .NET Core.

8.1.1. React SDK balstītie e-pakalpojumu konteineri

Uz React SDK balstītā e-pakalpojumu konteinera `Dockerfile` definīcijas piemērs:

```
# pull official base image
FROM node:lts-alpine as build

# add app
WORKDIR /buildDir
COPY ./webapp/examples-react-template .
COPY ./.env-react ./.env

# RUN npm and npm build
#RUN rm package-lock.json
RUN npm install
RUN npm run build

##########################################################
# Stage 2 - the production environment
FROM nginx:alpine
COPY                                  ./webapp/examples-react-template/nginx.conf
/etc/nginx/conf.d/default.conf
COPY --from=build buildDir/build /usr/share/nginx/public
COPY ./webapp/error /usr/share/nginx/public/
COPY docker-entrypoint.sh /usr/local/bin/docker-entrypoint.sh
RUN chmod +x /usr/local/bin/docker-entrypoint.sh

ENTRYPOINT ["/usr/local/bin/docker-entrypoint.sh"]
CMD ["nginx", "-g", "daemon off;"]
EXPOSE 80
```

Veikt konteinera `index.html` datnes konfigurāciju - dockerfile kodā jāiekļauj datni `docker-entrypoint.sh`, kas tiks izsaukta dokera veidošanas brīdī. REACT_APP_ ir atslēga, pēc kuras tiek veikta parametru meklēšana, to var aizstāt ar jebkuru citu brīvi izvēlētu konstanti.

```
#!/bin/sh

sed -i "s|{{PORT}}|${PORT:-8080}|g" /etc/nginx/conf.d/default.conf

IFS=$'\n'
for line in $(env); do
  key=$(echo $line | cut -d "=" -f 1)
  val=$(echo $line | cut -d "=" -f 2)
  sed -i "s|%REACT_APP_$key%|$val|g" /usr/share/nginx/html/index.html
done

exec "$@"
```

Savukārt, `index.html` satur no ārpuses konfigurējami vides mainīgi:

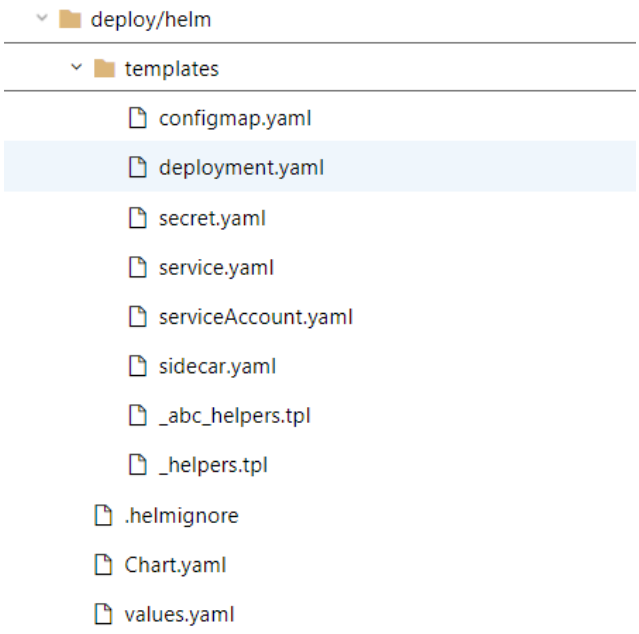
```
<script>
  window.env = {
    LOG_LEVEL:          '%REACT_APP_LOG_LEVEL%',
    AUTH_CLIENT_ID:     '%REACT_APP_AUTH_CLIENT_ID%',
    API_URL:            '%REACT_APP_API_URL%',
    AUTH_AUTHORITY:     '%REACT_APP_AUTH_AUTHORITY%',
  }
</script>
```

8.1.2. .NET core MVC balstītie e-pakalpojumu konteineri

.NET core MVC balstītie e-pakalpojumu konteineri tiek veidoti uz standarta (Add docker support…) Dockerfile definīcijas pamatā. E-pakalpojumu konteineru konfigurācijas notiek caur vides parametriem, kā tas ir paredzēts .NET core MVC realizācijā.

8.2. E-pakalpojumu Helm sriptu veidošana

E-pakalpojumu Helm skripti tiek piegādāti ārpus konteinera piegādes uz specializētu VRAA Nexus repozitoriju. Piegāde satur nepieciešamo Helm skriptu komplektu atbilstošā konteinera izmitināšanai VRAA vides K8s. Helm skriptus jāveido atbilstoši Helm 2 notācijai:



8.3. E-pakalpojumu platformas komponenšu konfigurēšana

E-pakalpojumu platformas komponenšu un vides konfigurācijas parametri tiek iznesti atsevišķā git repozitorijā un veidoti YAML valodā. YAML datnes netiek iekļautas konteinerā, bet piegādātas uz VRAA FTP un izmitinātas VRAA git repozitorijā. Pirms jaunu parametru izveides nepieciešams iepazīties ar jau esošajiem globālajiem vides parametriem un izmantot jau esošo parametru, ja tāds eksistē. Ja parametrs ir jauns tad ir jāizvērtē vai tas jāveido kā globālais vides mainīgais vai tas attiecās tikai uz konkrēto e-pakalpojumu un jāiekļauj komponentes līmeņa konfigurācijas datnē. Ja parametrs ir obligāts e-pakalpojuma izpildei tad tam nedrīkst norādīt noklusēto vērtību, tas nodrošina to ka aizmirstot šo parametru nodefinēt yaml datnē konteiners netiks startēts un administrators uzreiz uzzinās ka ir konfigurācijas kļūda. E-pakalpojuma frontend lietojuma konfigurācijas piemērs:

```
replicaCount: 1

image:
  repository: nexus.abc:5001
  pullPolicy: Always

config:
  assetsUrl: https://elieta-assets-dev-ic.abcsoftware.lv
  apiUrl: https://elieta-dev-ic.abcsoftware.lv/prlmonitoringaccount-api
  authClientId: elieta-prlmonitoringaccount-dev-ic
  authAuthority: https://elietaauth-dev-ic.abcsoftware.lv

resources:
  limits:
    cpu: "50m"
    memory: "100Mi"
  requests:
    cpu: "5m"
    memory: "10Mi"
```

Config sekcijā definētie mainīgie deployment laikā ir padoti uz docker image kā environment variable un izmantojot sh skriptu kas ir pašā konteinerī tiek embedoti index.html lapā.

Pārējās sekcijas ir kā override priekš eksistējošām vērtībām helm chartā, deployment laikā ir paņemts helm chart, kas ir apvienots ar vērtībām no šīs konfigurācijas + no environment specifiskās konfigurācijas (values-global.yaml). Vertības kas tiks padotas kā environment variables nosaka helm chart saturs (deployment.yaml).

8.3.1. E-pakalpojumu platformas globālie konfigurācijas parametri

Pirms jaunu parametru izveides nepieciešams iepazīties ar jau esošajiem globālajiem vides parametriem un izmantot jau esošo parametru ja tāds eksistē. E-pakalpojumu platformas globālie parametri:

Globālo parametru datne apskatāma https://git.vraa.gov.lv/lvp/eserviceplatform.examples/documents-etc.

```
image:
  repository: nexusrep.vraa.gov.lv

global:
  vpm:
    baseAddress: "https://vpmtest.vraa.gov.lv/LVP.STS/Default.aspx"
    metadataAddress:
"https://vpmtest.vraa.gov.lv/LVP.STS/FederationMetadata/2007-
06/FederationMetadata.xml"

  pfasAuth:
    baseAddress: "http://epakvisstv.vraa.gov.lv/STS/VISS.Pfas.STS"

  wso2:
    baseAddress: "http://apitestgw.vraa.gov.lv"

  searchEngine:
    baseAddress:
"http://ventabalancer.vraa.gov.lv/VISS.SearchEngine/WS/stable/SearchEngine.svc
"
    realm: "https://epakvisstv.vraa.gov.lv/VRAA.VISS2.SE/SearchEngine.svc"

  requestService:
    baseAddress:          "http://ausmatest8.vraa.gov.lv/Request.WebService/v1-
9/WcfService"
    anonymousEndpointAddress:
"http://ausmatest8.vraa.gov.lv/Request.WebService/v1-
9/WcfService/soap12WsAddresing"
    securedEndpointAddress:
"http://ausmatest8.vraa.gov.lv/Request.WebService/v1-
9/WcfService/ws2007FederationNoSct"
    realm: "URN:VISSTV:VISS.REQUEST.SERVICE"

  edk:
    baseAddress: "http://ventabalancer.vraa.gov.lv/VISS.EDK/WS2/stable"
    realm: "URN:TEST:VISS.EDK.WS2"

  notificationService:
    baseAddress: "http://ausmatest8.vraa.gov.lv"
    realm: "https://ivis.eps.gov.lv/NOT.WebService"

  addressFinder:
    baseAddress: "https://amktest.vraa.gov.lv/rest"

  eservicePlatform:
    assetsBaseAddress:                                    "https://eservices-
test.vraa.gov.lv/EservicePlatform.Assets"
    contextBaseAddress:                                   "https://eservices-
test.vraa.gov.lv/EservicePlatform.ContextApi"
    navigationBaseAddress:                                "https://eservices-
test.vraa.gov.lv/EservicePlatform.NavigationApi"
    gatewayBaseAddress: "https://eservices-test.vraa.gov.lv"
    idsBaseAddress: "https://eservices-test.vraa.gov.lv/Portal.IdentityServer"
    searchUrl: "https://lvptest.vraa.gov.lv/{language}/Meklesana"
    executedEserviceUrl:
"https://lvptest.vraa.gov.lv/{language}/KDV/E_pakalpojumu_saraksts"
    profileUrl: "https://lvptest.vraa.gov.lv/{language}/KDV/Profils"
    idleTime: "7"
```

```
    globalResourceUrl:
"https://epakvisstv.vraa.gov.lv/Lvp.EservicePlatform.Resources/Global/global.y
aml"
    breadcrumbs:
    - lv: "Sākums"
      ru: "Начало"
      en: "Home"
      link: "https://lvptest.vraa.gov.lv/{language}"
    - lv: "E-pakalpojumi"
      ru: "Э-услуги"
      en: "E-services"
      link: "https://lvptest.vraa.gov.lv/{language}/Epakalpojumi"
    - lv: "{eserviceName}"
      en: "{eserviceName}"
      ru: "{eserviceName}"
      link: "https://lvptest.vraa.gov.lv/{language}/Epakalpojumi/{eserviceId}"
    queryEncryptionKey: "SA54df1s6d1G#56sdf1we65wer65w1er"
    portalHomeUrl: "https://lvptest.vraa.gov.lv"

  mssql:
    datasource: "diana2.viss.int"
    host: "diana2.viss.int"

  rabbitmq:
    hosts:
      - host: venta1.viss.int
        port: 5672
    tls: false

  mongodb:
    hosts:
      - host: DAIRM2-N1.viss.int
        port: 27017
      - host: DAIRM2-N2.viss.int
        port: 27017
      - host: DAIRM2-N3.viss.int
        port: 27017
    replicaSet:
      enabled: true
      name: Wso2Dairm
```

8.3.2. E-pakalpojumu konfigurācija

Visi projekta darbībai nepieciešamie vides parametri pēc noklusējuma tiek definēti eservice-core index.js datnē. Ja ir nepieciešams pievienot papildus vides parametrus vai arī pārdefinēt eservice-core esošos, React projektā tiek izmantota vides parametru datne vai parametru konfigurēšana, izmantojot funkciju ConfigStore.set().

Norādot parametru vērtības ņemt vērā ka:

- Visa komunikācija, kas tiek veikta no pārlūka klienta datorā uz k8s izmitinātajiem resursiem jāveic izmantojot ārējās adreses.
- Visa komunikācija, kas tiek veikta no e-pakalpojuma servera koda uz k8s izmitinātajiem resursiem jāveic izmantojot iekšējās adreses.

- Resursiem, kas atrodas ārpus k8s ir jābūt norādītiem sidecar.

E-pakalpojumu ietvarā izmantojamās iekšējās adreses (visās vidēs ir vienādas), veidojas pēc šāda principa http://<svc-resursa-nosaukums>.<namespace>.svc.cluster.local:

- IDS - http://lvp-portal-identityserver.epak-system.svc.cluster.local;
- ContextApi - http://lvp-eserviceplatform-backend-contextapi.epak-system.svc.cluster.local;
- NavigationApi – http://lvp-eserviceplatform-backend-navigationapi.epak-system.svc.cluster.local;

Vides parametru datne

React projektam nepieciešamos vides parametru datus ir iespējams norādīt .env datnē. Piemēra parametri ir atrodami .env.example datnē. Ja eservice-core noklusējuma vides parametra nosaukums sakrīt ar lietotāja definēto parametru .env datnē, tad par aktuālo vērtību tiek ņemta lietotāja definētā vērtība .env datnē, un eservice-core esošā vērtība parametram tiek pārdefinēta uz jauno. Vides parametru datne tiek uzskatīta par galveno vietu, kurā definēt vai pārdefinēt projekta darbībai nepieciešamos parametrus.

Ņemot vērā, ka React projektam ne vienmēr ir nepieciešami visi parametri iekš vides parametru datnes, mapītes config datnē env.js ir definēts masīvs AVAILABLE_ENV, kurā ir jānorāda visi vides parametri, kuri tiek izsaukti tieši no React projekta komponentēm.

Vides parametru konfigurācija, izmantojot ConfigStore.set()

Projektā vides parametru definēšanai papildus var izmantot eservice-core ConfigStore.set() funkciju, kas ir pieejama React projekta index.js datnē. Pēc noklusējuma šeit tiek padotas visas lietotāja izveidotās funkcionālās komponentes e-pakalpojumam, bet to var arī izmantot vides parametru definēšanai. Ja šeit definētais vides parametra nosaukums sakrīt ar kādu no .env datnē definētā parametra nosaukumu, tad kā aktuālā vērtība tiek ņemta parametra vērtība, kas norādīta .env datnē.

Obligātie norādāmie vides mainīgie

Šobrīd e-pakalpojuma piemēru darbībai ir obligāti sekojošie vides mainīgie:

- ESERVICE_API_ENDPOINT_URL;
- ESERVICE_URN;
- ESERVICE_TITLE;
- ESERVICE_IS_ANONYMOUS;
- TRANSACTION_API_ENDPOINT_URL;
- INSTRUCTIONS_API_ENDPOINT_URL;
- NAVIGATION_API_ENDPOINT_URL;
- ASSETS_CDN_URL;
- AUTH_AUTHORITY_API_ENDPOINT;
- AUTH_CLIENT_ID;
- AUTH_CLIENT_SECRET;
- ASSETS_CDN_URL;

ESERVICE_TITLE un ESERVICE_IS ANONYMOUS eservice-core ir definētas noklusējuma vērtības, tādēļ šo mainīgo nenorādīšana neietekmēs projekta veiksmīgu darbību.

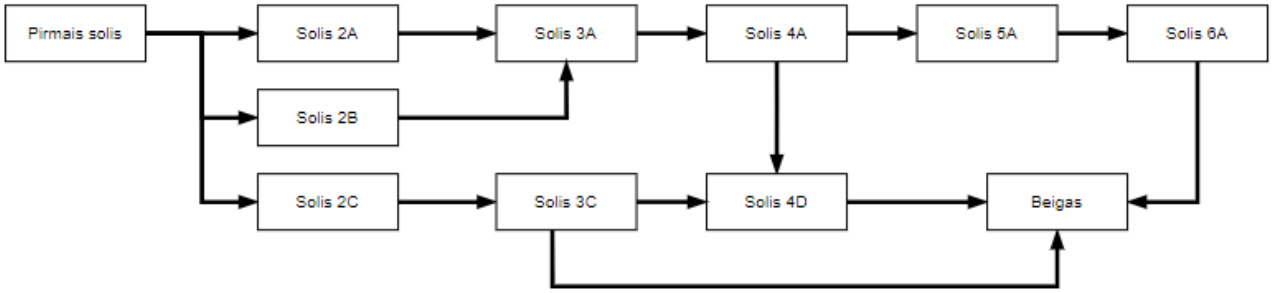
9. E-paklpojumu piemēri

Piemēri demonstrē e-pakalpojumiem pieejamo bibliotēku un pieeju iespējas.

ComplexUI

Kompleksā dizaina e-pakalpojumu piemēri, kas satur dizainu SDK un gatavus komponenšu lietojumus. Tādus kā – Ievades laukus, validācijas, datu režģus, kalendāra lietojumu, u.c.

ComplexUI ir sarežģītākais piemērs, kurā attēlota soļu pāreja pēc piemēra grafa:



EP500

E-pakalpojuma piemēri attēlo talona izmantošanu (lietotāja pilnvarošanu) un IVIS pieprasījuma sūtīšanu.

Service Integration

Servisa integrācijas e-pakalpojuma piemēri attēlo servisu pieprasījumus, ContextAPI un PaymentApi lietojumus. Servisu veidi un detaļas aprakstītas 7.nodaļā.

Template

E-pakalpojuma šablonu piemēri paredzēti kā bāzes izejas punkts jaunam e-pakalpojumam.

10. Biežāk sastopamās problēmas un to risinājumi

1. CORS

Simptoms: Caur AJAX pieprasot kādu resursu pārlūka konsolē parādās kļūda – “XMLHttpRequest cannot load <vēlamā adrese uz resursu>. No 'Access-Control-Allow-Origin' header is present on the requested resource. Origin <adrese no kuras pieprasa> is therefore not allowed access.”.

Risinājums: Kļūda nozīmē, ka serveris, kuram pieprasa resursu nedrīkst to atgriezt uz šo konkrēto adresi no kuras tas tiek pieprasīts. Lai atrisinātu šo kļūdu ir iespējami vairāki risinājumi:

- Mainīt adresi no kuras pieprasa
- Mainīt CORS iestatījumus serverim, lai tie atļautu konkrēto adresi.

Līdzīgas situācijas var gadīties arī ar tādiem pieprasījuma parametriem, kā pieprasījuma galvenes dati un pieprasījuma metodes(GET, POST)

2. E-pakalpojumu piemēri - react

Simptoms: Lokāli darbinot e-pakalpojuma piemēra react daļu, tas nebūvējās(ar kļūdu Couldn't import the config file at ../nwb.config.js:) vai uzbūvējoties neparāda galveni un kājeni.

Risinājums: pārliecinieties, ka .env-react, .env un .env-bff faili ir korekti aizpildīti(ir pareizie ceļi uz resursiem) un atrodas pareizajās direktorijās:

- Darbinot ar docker – env failiem jābūt root direktorijā
- Darbinot caur npm – .env-react fails jāpārkopē uz konkrēto react direktoriju un jāpārsauc par .env

3. Storybook

Simptoms: Būvējot storybook, tas neizdodas un tiek izdota kļūda: “*FATAL ERROR: Ineffective mark-compacts near heap limit Allocation failed - JavaScript heap out of memory*“

Risinājums: Problēma ir saistīta ar TerserPlugin paciņu, kuru storybook izmanto lai saspiestu izejas koda failus un uzģenerētu to kartes. Ir divi variant, kā to risināt:

- Palielināt NPM un Node atļauto izmantojamo atmiņu, kopumā vai konkrētajam procesam.
- Izslēgt TerserPlugin minimizāciju .env failā norādot mainīgo STORYBOOK_WEBPACK_MINIMIZE kā false

4. E-pakalpojumu piemēri - servisu izsaukumi

Simptoms: Darbinot e-pakalpojumu tiek saņemts 404 vai 500(Invalid URI provided) kļūdas paziņojums, bet pats e-pakalpojums darbojās. Šī problēma attiecās arī uz izmantotajiem bilžu assetiem

Risinājums: E-pakalpojumi izmanto daudz servisus, kur katram ir savādāka adrese, kura var arī mainīties atkarībā no vides, līdz ar ko katrai komponentei ir konfigurācijas faili, kurus vajadzētu apskatīt un pārkonfigurēt. Darbinot e-pakalpojumus ar Docker Compose konfigurācijas faili attiecīgā repozitorija saknes direktorijā. Konfigurācijas faili var būt šādi(**Mainot konfigurācijas failus ir jāatjauno arī to piemēru faili!**):

- .env – konfigurācijas fails, kuru pēc noklusējuma pieņem Docker compose, tajā galvenokārt ir portu un adrešu konfigurācijas priekš attiecīgo Docker konteineru uzbūvēšanas. HTMLSDK un REACTSDK gadījumos satur arī mainīgos priekš pašu SDK būvēšanas, kā arī Storybook.

- .env-react – konfigurācijas fails epak piemēru react repozitorijiem. Galvenokārt satur servisu adreses uz kurām react izsūta pieprasījumus. Ja ir vairāki e-pakalpojumi vienā projektā, tad būtu nepieciešams pārliecināties, ka katrs izmanto tam paredzētu apstrādes servisu
- .env*.example – piemēra konfigurācijas fails, kuru pirmajā reizē lejupielādējot ir jāpārsauc noņemot .example un jānokonfigurē atbilstoši videi.
- appsettings.Example.json – konfigurācijas fail .Net komponentēm, kuru nepieciešams pārkopēt uz appsettings.Production.json vai atbilstošo vidi, kādu izmantojat un nokonfigurēt. Satur .Net vides parametrus, kā arī servisu adreses un versijas.

Ja tiek darbināti Helm skripti, tad darbināšanai nepieciešams modificēt un padod konfigurāciju failus. Konfigurācijas maiņas gadījumā nepieciešams atjaunot arī Helm skriptos padodamās konfigurācijas un piemēru failus.

5. E-pakalpojumu piemēri – ReactSDK CDN

Simptoms: konsolē parādās šādi ziņojumi:

- Uncaught SyntaxError: Unexpected token '<'
- Uncaught ReferenceError: ReactSDK is not defined

Risinājums: Pārbaudiet vai SDK CDN ceļš ir pareizs un vai CDN darbojās, ceļš parasti ir šāds:

CDNBaseAddress/CDNVersion/SDK/ReactSDK/js/controls-react.min.js

6. Servisu izsaukumi – x-milestoneId

Simptoms: Parametra 'x-milestoneId' vērtība ('URN:IVIS:100001:EP.VISS-EP00-v1-0-MS-CallCalcSync') nav korekta: neatbilst aktuālajai transakcijai ('URN:IVIS:100001:EF.VISS-EF00-v1-0-TR-17233').

Risinājums: Pārliecinieties, ka priekš x-milestoneId tiek izmantots e-pakalpojuma izpildes robežpunkta identifikators('URN:IVIS:100001:EF.VISS-EF00-v1-0-TR-17233) nevis lietotāja e-pakalpojuma transakcijas Nr ('URN:IVIS:100001:EP.VISS-EP00-v1-0-TR-17233) – Atšķirība šajā gadījumā ir 2 simbolos. E-pakalpojumu piemēros šī ir eServiceId vērtība konfigurācijas failos