**ESPBI IS modernizavimo (dekomponavimo) projekto programinė įranga ir programinės įrangos konfigūravimo paslaugos**

Table of contents

1. PLANUOJAMA PLATFORMOS ARCHITEKTŪRA 3

1.1. Aprašas 3

1.2. Principai 4

1.2.1. Viskas kaip kodas 4

1.2.2. "GitOps" 4

1.2.3. DevSecOps 5

1.2.4. Saugus pagal dizainą 5

1.2.5. Stebimumas 5

1.3. Reikalingi sprendimai 5

2. PROGRAMINĖ ĮRANGA, ĮRANKIAI IR PASLAUGOS 7

2.1. Reikalavimai konteinerizavimo platformai 7

2.2. Platformos diegimo reikalavimai 9

2.3. Platformos palaikymo reikalavimai 10

2.4. Reikalavimai pirminio kodo valdymo sistemai 10

2.5. Nuolatinio integravimo ir nuolatinio diegimo (CI/CD) valdymo reikalavimai 11

2.5.1. Reikalavimai komandų grandinėms (angl. pipelines) 12

2.5.2. Reikalavimai CI/CD įrankiams 13

2.5.3. Reikalavimai aplikacijos saugumo testavimo įrankiams 15

2.6. Konteinerių registro reikalavimai 15

2.7. Artefaktų registro reikalavimai 16

2.8. Automatizuoto infrastruktūros diegimo ir valdymo reikalavimai 16

2.9. Reikalavimai įrankiams 17

2.10. Platformos Priežiūros paslaugų reikalavimai 18

**Schemų sąrašas**

[Pav. 1. Platformos komponentų architektūra 3](#_Toc194066384)

# PLANUOJAMA PLATFORMOS ARCHITEKTŪRA



Pav. 1. Platformos komponentų architektūra

## Aprašas

1. Sprendimas bus įdiegtas per du valstybės įmonės „Registrų centras“ (toliau – Perkančioji organizacija) duomenų centrus, kad būtų užtikrintas aukštas patikimumas. Aplikacijos turi būti diegiamos ir vykdomos konteinerių klasteriuose. Aplikacijų duomenų bazės turi būti įdiegtos esamuose „IBM Power E1080“ serveriuose, prijungtuose prie replikuojamos SAN tipo duomenų saugyklos. Kiekviename duomenų centre diegiami aukšto patikimumo konteinerių klasteriai („OpenShift“ arba lygiaverčiai), susidedantys iš 3 master narių, kurie turi būti įdiegti Perkančiosios organizacijos virtualizacijos platformoje, ir mažiausiai 3 worker narių, kurie turi būti įdiegti į aparatinę įrangą (angl. Bare-metal).
2. Worker nariuose turi būti įdiegta ši programinė įranga:
   1. Programiškai valdoma duomenų saugykla, kuri suteikia išliekančiuosius diskus aplikacijoms, veikiančioms konteinerių klasteryje;
   2. E.Sveikatos (ESPBI) demo aplikacija – aprašyta Platformos diegimo reikalavimuose.
3. Turi būti įdiegti trys konteinerių klasteriai: du PROD kiekviename duomenų centre (A), vienas vystymo aplinkai (B) pagrindiniame duomenų centre. Klasterio narių (B) fiziniai ištekliai gali būti mažesni nei A tipo (PROD). PROD aplinką sudaro 3 worker nariai kiekviename duomenų centre, nePROD (DEV/TEST/STAGE/QAS ar panaši) aplinkų klasteris yra viename duomenų centre ir susideda iš 3 worker narių. Tolesnis klasterių pajėgumų didinimas galėtų būti atliekamas pridedant po 3 narius.
4. Konteinerių platformos valdymo sistemos turi būti įdiegtos esamoje virtualizacijos platformoje, pageidautina konteinerių klasterio diegime (OpenShift arba lygiavertis), siekiant užtikrinti, kad valdymo įrankiai būtų visada prieinami.
5. Duomenų, rašomų į aplikacijos konteinerio nuolatinę saugyklą, repliką tarp duomenų centrų užtikrina aplikacijos komponentai (ESPBI arba e.Sveikata). Aplikacijų duomenys, įrašyti į atskirą aparatinę objektų saugyklą, replikuojami saugyklos priemonėmis. Aukštą aplikacijos pasiekiamumą viename duomenų centre užtikrina konteinerių klasteris.
6. Sprendimas turi būti suprojektuotas ir licencijuotas taip, kad galėtų dirbti bent 50 programuotojų, užtikrinant pakankamus skaičiavimo resursus, sistemos plečiamumą ir našumą, kad vienu metu būtų galima vykdyti bent 10-ies aplikacijų, vystomų 50 git saugyklose, vystymo veiklą, testavimą ir efektyvų diegimą.
7. Tiekėjas privalo pateikti visas licencijas, (tame tarpe ir gamintojo palaikymo licencijas), kurios galiotu 60 mėnesių, perkančiajai organizacijai per 3 mėnesius nuo sutarties pasirašymo dienos. Jei tiekėjas teikia prenumeratos tipo licencijas, jos turi apimti visus reikalavimus ir sąlygas, taikomus gamintojo palaikymui.
8. Tiekėjas turi užtikrinti, kad Paslaugos atitinka Kibernetinio saugumo reikalavimų apraše, patvirtintame Lietuvos Respublikos Vyriausybės 2018 m. rugpjūčio 13 d. nutarimu Nr. 818 „Dėl Lietuvos Respublikos kibernetinio saugumo įstatymo įgyvendinimo“, nurodytus reikalavimus.

## Principai

### Viskas kaip kodas

1. Platforma turi palaikyti infrastruktūros kaip kodo (IaC) galimybes, leidžiančias apibrėžti ir valdyti infrastruktūros išteklius, pvz., skaičiavimą, saugyklą ir tinklą, naudojant žmogaus skaitomus (angl. Human-readable) konfigūracijos failus, tokius kaip YAML arba JSON. Tai leistų vieningai valdyti visą programos gyvavimo ciklą, įskaitant kompiliavimą, diegimą ir vykdymo konfigūraciją, naudojant kodą. Platforma turi palaikyti kelis IaC įrankius, tokius kaip Terraform arba Ansible, kad suteiktų lankstumą ir galimybę rinktis. Todėl visi platformos komponentai, įskaitant aplikacijas, paslaugas (angl. service) ir infrastruktūrą, turi būti apibrėžti ir valdomi kaip kodas, užtikrinant versijų valdymą, patikrinamumą ir atkuriamumą.

### "GitOps"

1. Platforma turi turėti galimybę būti integruota su "Git" pagrįstomis versijų valdymo sistemomis, kad būtų galima valdyti visą programos gyvavimo ciklą, įskaitant kompiliavimą, diegimą ir vykdymo konfigūraciją. Tai įgalintų "GitOps" darbo eigas, leidžiančias automatiškai įdiegti aplikacijas ir paslaugas, pagrįstas "Git push" įvykiais, “pull” užklausomis ar kitais su "Git" susijusiais paleidikliais. Platforma turi užtikrinti sklandžią integraciją su nuolatinio integravimo / nuolatinio diegimo (CI / CD) komandų grandinėmis, kad kodo pakeitimai būtų automatiškai kompiliuojami, testuojami ir diegiami gamyboje. Git pirminio kodo saugykla turėtų būti pagrindinis tiesos šaltinis deklaratyviai klasterių, aplinkų ir jose įdiegtų aplikacijų konfigūracijai. Todėl visi aplikacijos ar infrastruktūros pakeitimai bus sekami, bus galimybė juos tikrinti ir valdyti versijas, pateikiant aiškią pakeitimų ir atšaukimų istoriją.

### DevSecOps

1. Platforma turi galėti integruoti saugumą į vystymo procesą, užtikrindama, kad saugumas būtų prioritetas CI / CD komandų grandinėje. Turi palaikyti saugumo testavimą, pvz., pažeidžiamumų skenavimą, statinius ir dinaminius aplikacijų skanavimus, kurie yra būtini siekiant užtikrinti, kad aplikacijos ir paslaugos būtų saugios. Platforma turi teikti realaus laiko grįžtamąjį ryšį apie aplikacijos saugumą programuotojams, kad saugumas nuo pat pradžių būtų integruotas į vystymo procesą. Šis metodas reikalingas siekiant užkirsti kelią saugumo spragoms ir sumažinti saugumo pažeidimų riziką, įtraukiant saugumo testus ir grįžtamąjį ryšį į CI / CD komandų grandinę, suteikiant galimybę programuotojams aptikti ir spręsti saugos problemas ankstesniame vystymo proceso etape.

### Saugus pagal dizainą

1. Platforma turi būti suprojektuota atsižvelgiant į saugumą, įtraukiant geriausias saugos praktikas, pvz., mažiausių reikiamų privilegijų prieigą, saugius ryšio protokolus ir saugų duomenų saugojimą. Įtaisytosios (angl. Built-in) saugos funkcijos, pvz., tinklo politikos, paslapčių valdymas ir rolėmis pagrįstas prieigos valdymas (RBAC), reikalingos siekiant užtikrinti, kad aplikacijos ir paslaugos būtų saugios pagal nutylėjimą. Platforma turi turėti galimybę būti integruota su esamais saugos įrankiais ir paslaugomis, tokiomis kaip tapatybės ir prieigos valdymo (IAM) sistemos, kad būtų užtikrinta vieninga saugumo pozicija. Taip pat platforma turi gebėti teikti beveik realaus laiko saugumo stebėjimo ir įspėjimo galimybes, užtikrinančias, kad saugumo incidentai būtų greitai aptikti ir į juos būtų reaguojama.
2. Platforma turi palaikyti nulinio pasitikėjimo saugos modelį, kai visas gaunamas ir išeinantis srautas yra autentikuojamas, autorizuojamas ir šifruojamas, neatsižvelgiant į siuntėją ar adresatą. Detalus prieigos valdymas yra būtinas siekiant užtikrinti, kad kiekvienas komponentas, aplikacija ar paslauga turėtų mažiausiai prieigos teisių, reikalingų veikimui užtikrinti. Todėl visi perduodami ar saugomi duomenys turi būti užšifruoti ir apsaugoti nuo neautorizuotos prieigos.

### Stebimumas

1. Platforma turi teikti stebėjimo ir žurnalų įrašų registravimo realiuoju laiku galimybes, užtikrindama, kad visi komponentai, aplikacijos ir paslaugos ir jų parametrai yra stebimi ir analizuojami. Norint užtikrinti visavertį, vieningą ir pilną sistemos vaizdą, reikia naudoti trasavimo, stebėjimo ir žurnalų įrašu priemones. Platforma turi teikti įspėjimų ir pranešimų siuntimo galimybes, kad problemos būtų greitai aptiktos ir į jas būtų reaguojama.

## Reikalingi sprendimai

1. Aplikacijų veikimo tęstinumui ir saugumui užtikrinti planuojami tokie sprendimai:
   1. turėtų būti automatizuotai tikrinama pirminio kodo kokybė;
   2. aplikacijų pirminis kodas ir susiję objektai turėtų būti tikrinami dėl galimų pažeidžiamumų;
   3. aplikacijų kūrimas iš pirminio kodo (angl. build) turėtų būti automatizuotas ir saugus;
   4. aplikacijos versijų ir laidų (angl. release) valdymas turėtų būti kontroliuojamas;
2. Sprendimo funkcionalumui užtikrinti Perkančioji organizacija suteikia prieigą prie šių komponentų:
   1. centralizuotos žurnalų įrašų saugojimo sistemos;
   2. centralizuotos metrikų saugojimo sistemos;
   3. atsarginių kopijų valdymo įrankių;
   4. autentikavimo posistemių;
   5. DNS tarnybos;
   6. pirminio kodo patikros įrankių.
3. Taip pat sprendimo funkcionalumui užtikrinti, Perkančiosios organizacijos virtualizacijos platformoje, turi būti sudiegti papildomi konteinerizacijos platformos valdymo komponentai:
   1. Git tipo pirminio kodo valdymo sistema;
   2. Automatizuoto infrastruktūros diegimo ir valdymo sistema;
   3. Konteinerio atvaizdų (angl. container image) registras;
   4. Aplikacijų tiekimo grandinės patikros įrankiai;
   5. Aplikacijų nuolatinio integravimo, pateikimo ir diegimo (CI/CD) įrankiai;
   6. Privilegijuotų paskyrų prieigos valdymo sistemos;
   7. Aplikacijų jautrios informacijos saugojimo priemonės (angl. secret store/vault).

# PROGRAMINĖ ĮRANGA, ĮRANKIAI IR PASLAUGOS

1. Turi būti įdiegtas ir sukonfigūruotas visapusiškas Platformos įrankių rinkinys, aplikacijų vystymui, diegimui ir valdymui. Platformos įrankiai turi būti suprojektuoti ir įdiegti taip, kad atitiktų šiuos nefunkcinius reikalavimus:
   1. Užtikrintų minimalias prastovas ir aukštą patikimumą, kai veikimo laikas yra 99,5 % ar didesnis, svarbiausioms teikiamoms viešosioms paslaugoms.
   2. Resursų plečiamumą horizontaliai ir vertikaliai, kad patenkintų augimo poreikius ir galėtų aptarnauti padidėjusius srautus ir užklausų kiekius.
   3. Užtikrintų Platformos saugumą ir vientisumą, taikant patikimas prieigos kontrolės priemones, šifravimą ir audito mechanizmus, siekiant apsaugoti jautrius duomenis ir išlaikyti visuomenės pasitikėjimą.
   4. Optimizuotų našumą, teikdami greitą atsakymo laiką ir efektyvų užklausų apdorojimą, taip užtikrinant viešųjų paslaugų prieinamumą.
   5. Būtų lengvai eksploatuojami, teikdami paprastus atnaujinimus, pataisymus ir trikčių šalinimo galimybes, kad sumažintų prastovų laiką ir užtikrintų viešųjų paslaugų tęstinumą.

## Reikalavimai konteinerizavimo platformai

1. Turi būti pateiktos reikalingos nuolatinio galiojimo (angl. Perpetual) licencijos arba licencijų prenumeratos (angl. Subscription), skirtos licencijuoti serverius, kurių kiekvienas turi iki 128 branduolių (angl. cores) per vieną arba du fizinius procesorius.
2. Turi būti pateiktos reikalingos nuolatinio galiojimo (angl. Perpetual) licencijos arba licencijų prenumeratos (angl. Subscription), skirtos licencijuoti virtualų Platformos valdymo klasterį, kuris apims bent 24 virtualius procesorius (angl. vCPU).
3. Perkančioji organizacija įsipareigoja nupirkti reikiamas licencijas serveriams.
4. Turi turėti šias arba lygiavertes savybes ir funkcionalumus:
   1. Aukšto patikimumo;
   2. Saugos;
   3. Rezervinio kopijavimo;
   4. CI/CD;
   5. GitOPS;
   6. Infrastruktūra kaip kodas;
   7. Automatizuoto diegimo;
   8. Automatizuoto konfigūracijos valdymo;
5. Komunikavimo (angl. service mesh);
   1. Stebėjimo/monitoringo;
   2. Programinio kodo valdymo;
   3. API.
6. Platforma turi palaikyti konteinerių (pvz. Docker ar lygiaverčių) bei jų galimybes praplečianti (pvz. Kubernetes ar lygiavertį) sprendimą.
7. Platforma turi palaikyti integracijas su labiausiai paplitusiais konteinerių vaizdų registrais (angl. Image Repository): Docker Hub, Gooogle Container Registry, IBM Cloud Container Registry, Microsoft Azure Container Registry, Red Hat Container Registries ir kitais, palaikančiais standartizuotas žiniatinklio paslaugas kaip Docker Registry HTTP API ar lygiaverčiais registrais ir papildomai:
   1. turi būti galimybė vienu metu Platformoje naudoti neribotą skaičių registrų;
   2. turi būti galimybė naudoti viešai prieinamus ir vidinius (angl. On Premise) registrus;
8. turi būti galimybė konteinerių kūrimo metu panaudoti registruose esančius atvaizdus (angl. Container image);
   1. turi būti galimybė apriboti galimų registrų ar tuose registruose naudojamų atvaizdų sąrašą (angl. Whitelist) ir panaudojimą Platformoje.
9. Platformoje turi būti galimybė, per naudotojo sąsają, administruoti techninėje specifikacijoje numatytą skaičių „Kubernetes“ ar lygiaverčių konteinerių klasterių:
   1. aktyviu režimu (angl. Online) keisti Konteinerio klasterių pajėgumus papildomai pridedant naują Konteinerio klasterio mazgą (angl. Node) arba ji pašalinant;
10. naikinti, kurti ir priskirti Konteinerio klasteriui neribotą skaičių bendrų duomenų sluoksnių (angl. Persistent storage);
11. naikinti, pridėti prie Konteinerio klasterio neribotą mazgų (angl. Node) srauto paskirstymo sluoksnius (angl. Balancers);
    1. aktyviu režimu (angl. Online) valdyti Konteinerių klasteryje esančios paslaugos (angl. Services) (toliau - Paslauga) pajėgumus, pridedant arba mažinant paslaugos mazgų (angl. Node) skaičių;
    2. naikinti, kurti neribotą skaičių Konteinerių klasterio veikimui reikalingų vidinių Konteinerių klasterio tinklų;
    3. stebėti Konteinerių klasterio, jo mazgų (angl. Nodes), duomenų sluoksnių (angl. Persistent storage), srauto paskirstymo sluoksnių (angl. Balancers), Konteinerių klasteryje esančių Paslaugų, individualaus konteinerio veikimo parametrus kaip: CPU, RAM, HDD (bet neapsiribojant jais) ir Konteinerių klasteriui veikti sukurtus ir reikalingus tinklo apkrovos parametrus.
12. Platforma turi turėti vartotojo sąsają, galinčią valdyti saugos parametrus neribotam skaičiui "Kubernetes" arba lygiaverčių konteinerių klasterių:
    1. Vertinti ir formuoti ataskaitas apie atitiktį labiausiai paplitusiems standartams: NIST, CIS ir kitiems;
    2. Aptikti ir įvertinti saugumo rizikas;
    3. Užtikrinti saugos politikų vykdymą;
    4. Tikrinti atvirus konteinerių prievadus;
    5. Skenuoti konteinerio atvaizdus ir konteinerių klasterio narius ieškant žinomų pažeidžiamumų;
    6. Turi palaikyti pažeidžiamumų šaltinius, tokius kaip: Debian Security Tracker, Ubuntu OVAL, Photon OVAL, RedHat OVAL ir kitus.
13. Platforma turi automatiškai pagal aprašytas taisykles paskirstyti konteinerių vykdymą tarp konteinerių klasterio narių.
14. Platformoje turi būti galimybė kurti, naikinti neribotą kiekį Linux ir Windows OS virtualių serverių bei valdyti jiems priskirtus resursus: CPU, RAM, HDD.
15. Platforma turi palaikyti rankinio ir automatinio Paslaugos diegimo (angl. Continuous deployment) principus. Turi būti galimybė šiuos procesus vykdyti Platformos ribose. Diegimus Platformoje atlikti naudojant programinio kodo, kaip GitLab ar lygiavertėmis technologijomis/įranga paremtas Git programinio kodo saugojimo principus, saugykla ( angl. repository).
16. Platforma turi gebėti veikti debesijos tiekėjų paslaugose, tokiose kaip Amazon Web Services, Google Cloud Platform, Microsoft Azure ir pan. ir gebėti palaikyti ne mažiau kaip 2 (du) sudubliuotus duomenų centrus (DC) bet kurioje hibridinėje kombinacijoje:
    1. Cloud ßà on Premise;
    2. Cloud ßà Cloud;
    3. On Premise ßà on Premise.
17. Platformos administravimo sritis turi palaikyti rolių atskyrimą pagal galimo funkcionalumo naudojimą, pvz:
    1. Administratoriaus rolė — visiška Platformos kontrolė, produkcinio ir neprodukcinio projekto, Konteinerių klasterio ir kito komponento administravimas;
    2. Programuotojo rolė — galimybė naudotis savitarnos portalu (angl. Self-service) ir savarankiškai administruoti neprodukcinio lygio jam priskirtą projektą arba infrastruktūros aplinką;
    3. Skaitytojo rolė — leidžianti peržiūrėti, bet ne modifikuoti (angl. Read only) Platformos konfigūraciją ir kitą informaciją.
18. Platforma turi gebėti palaikyti tiek paslaugas (angl. PaaS) ir aplikacijas (angl. SaaS) skirtas debesijai, tiek standartinėms aplinkoms, veikiančioms vidiniuose duomenų centruose (angl. on Premise), tiek ir hibridines aplikacijas, skirtas abiejų tipų sprendimams.
19. Platformoje turi būti įgalinta platinti bendras saugumo, kitų nustatymų taisykles visuose arba pasirinktoje dalyje valdomų Konteinerių klasterių.
20. Platforma turi gebėti apjungti ir valdyti neribotus fizinius resursus (tokius kaip CPU, HDD, RAM) tiek skaičiumi, tiek bendra apimtimi.
21. Platforma turi suteikti galimybę kurti ir naudoti iš anksto paruoštas ir administruojamas (angl. Managed services) paslaugas kaip: SaaS, PaaS, IaaS.
22. Turi veikti x86 platformos serveriuose.
23. Platforma turi palaikyti bendro duomenų sluoksnio (angl. Persistent storage) formavimo programinę įrangą kaip Ceph arba lygiavertę, kuri leistų formuoti duomenų saugojimo sluoksnį tiek iš atskirų serverių diskų, tiek iš SAN tinkle esančių blokinių saugyklų. Duomenų saugojimo sluoksnio kūrimo programinė įranga turi gebėti palaikyti tiek blokinį (angl. Block storage), tiek failų (angl. File system) saugyklos tipą.
24. Turi būti galimybė Platformoje esantiems Konteinerių klasteriams pritaikyti automatizuotą apkrovos ir valdymo resursų orkestravimą. Šis orkestravimas turi apimti tiek Konteinerių klasterio mazgus (angl. Node), tiek Konteinerių klasteryje veikiančias paslaugas (angl. Services). Turi būti galimybė numatyti galimas minimalias ir maksimalias resursų naudojimo arba plėtimosi ribas. Turi būti galimybė apriboti infrastruktūros išteklių (CPU, RAM, Disk) panaudojimą, apibrėžiant išteklių kvotas.
25. Turi būti galimybė L3 ir L4 lygiuose valdyti (drausti ar leisti) duomenų srautus tarp konteinerių, naudojant aprašytas politikas.
26. Turi būti galimybė netrikdant Platformos veikimo atlikti tiek Platformos, tiek Platformos komponentų ir serverių atnaujinimus.
27. Leisti loginį bei virtualų prod/pre-prod/test/dev aplinkų atskyrimą.

## Platformos diegimo reikalavimai

1. Platforma turi būti įdiegta per 6 mėnesius nuo sutarties pasirašymo.
2. Platformos įdiegimas patvirtinamas pasirašant priėmimo-perdavimo aktą.
3. Iki diegimo tiekėjas turi atlikti Perkančiosios organizacijos infrastruktūros ir poreikių analizę, kurios metu gali būti pateikiamos rekomendacijos dėl platformos diegimui numatytos Perkančiosios organizacijos infrastruktūros.
4. Tiekėjas iki diegimo pradžios turi paruošti architektūrinį sprendimą bei parengti šio sprendimo schemas bei dokumentaciją. Architektūrinis sprendimas turi užtikrinti aukštą platformos prieinamumą.
5. Tiekėjas iki diegimo pradžios turi su Perkančiąja organizacija susiderinti diegimo planą su konkrečiais diegimo etapais bei terminais.
6. Diegimas atliekamas tik gavus raštišką Perkančiosios organizacijos sutikimą (raštu ar el. paštu).
7. Tiekėjas turi sukonfigūruoti integracijas su CI / CD įrankiais, konteinerių registru, vartotojo sąsaja, rodančia visus sprendime įdiegtus konteinerių klasterius, ir jų pažeidžiamumų ataskaitas. Sukonfigūruoti bendro duomenų sluoksnio (angl. Persistent storage) formavimo programinę įrangą. Sukonfigūruoti integracijas su įsigyta ir esama infrastruktūra: DNS, NTP, SSO, Srauto paskirstymo įranga).
8. Priėmimo perdavimo aktas pasirašomas po platformos įdiegimo ir bandomosios eksploatacijos periodo (toliau — Bandomoji eksploatacija). Bandomosios eksploatacijos metu, bet ne ilgiau kaip 1 mėn., tiekėjas su Perkančiąja organizacija raštu suderina bandymų scenarijus, apimančius automatizuotą aplikacijos diegimą ir dubliuotų platformos komponentų patikimumą gedimo atveju.
9. Bandymo eksploatacijai visas reikalingas priemones parūpina Perkančioji organizacija.
10. Bandomoji aplikacija turi tenkinti minimalaus funkcionalumo principus ir turi apimti šias Platformos galimybes:
    1. bandomoji aplikacija turi būti diegiama iš kodo saugyklos — GitLab ar lygiavertės;
    2. bandomoji aplikacija turi veikti klasteriniu režimu per visus klasterio mazgus;
11. bandomojoje aplikacijoje turi būti panaudotas bendras Platformos duomenų sluoksnis (angl. Persistent storage);
12. bandomojoje aplikacijoje turi būti panaudotas Platformos srauto paskirstymo sluoksnis (angl. Balancers);
    1. bandomojoje aplikacijoje turi būti panaudota duomenų bazė (kaip MySQL ar lygiavertė).
13. Bandomąją aplikacija pasirūpina Perkančioji organizacija.

## Platformos palaikymo reikalavimai

1. Platformos programinės įrangos licencijos būti įdiegtos per 6 mėnesius nuo sutarties pasirašymo.
2. Teikiamas palaikymas pagal platformos gamintojo standartines palaikymo sąlygas, kurios:
   1. suteikia Perkančiajai organizacijai galimybę naudotis naujausiomis siūlomos programinės įrangos versijomis be papildomo mokesčio;
3. užtikrina teisę į programinės įrangos klaidų taisymus (angl. Bug-fix);
   1. užtikrina naujų versijų ir dokumentacijos atsisiuntimą iš gamintojo internetinio tinklapio be papildomo mokesčio;
   2. suteikia galimybę naudotis programinės įrangos gamintojo internetiniu portalu;
   3. užtikrina informacijos apie naujausią programinę įrangą gavimą;
   4. suteikia galimybę Perkančiajai organizacijai tiesiogiai kreiptis į gamintoją 24/7;
   5. suteikia visus kitus papildomus privalumus bei funkcijas pagal teikiamo gamintojo palaikymo lygį.

## Reikalavimai pirminio kodo valdymo sistemai

1. Tiekėjas turi suprojektuoti pirminio kodo valdymo sistemą pagal šiuos reikalavimus:
2. Turi būti suprojektuota gamintojo palaikoma Git tipo pirminio kodo valdymo sistema.
3. Git priminio kodo valdymo sistema turi būti pasiekiama tokiais protokolais:
   1. Git per SSH, naudojant privatų raktą tapatumo nustatymui.
   2. Git per HTTPS, naudojant naudotojo vardą ir slaptažodį tapatumo nustatymui.
4. Naudotojo sąsaja turi būti realizuota saityno priemonėmis bei suteikti naudotojui galimybę:
   1. Valdyti (kurti, keisti, trinti) SSH raktus, naudojamus prisijungimui prie pirminio kodo valdymo sistemos.
   2. Keisti pirminį kodą ir pateikti (angl. Commit) pakeitimus. Pirminio kodo redaktorius turi palaikyti Java, JavaScript, YAML sintaksės paryškinimą (angl. syntax highlight).
   3. Padaryti pirminio kodo saugyklos (angl. repository) kopiją (angl. fork ).
   4. Atlikti failų paiešką pirminio kodo valdymo sistemoje.
   5. Atlikti tekstinę paieška failuose, saugomuose pirminio kodo valdymo sistemoje.
   6. Išsiųsti pirminio kodo sujungimo užklausą (angl. pull request) su aprašymu.
   7. Sukurti ir naudoti pirminio kodo sujungimo užklausų aprašymo šablonus.
   8. Peržiūrėti sujungimo užklausų konfliktus ir nurodyti kurią pirminio kodo versiją naudoti.
5. Turi būti realizuota API sąsaja arba komandinės eilutės priemonė, leidžianti automatizuoti:
   1. Pirminio kodo saugyklos (angl. repository) kūrimą, keitimą ir trynimą.
   2. Pirminio kodo šakos (angl. branch) ir žymos (angl. tag) kūrimą ir trynimą.
   3. Saityno gaudyklės (angl. webhook) parametrų keitimą.
6. Turi realizuotus programinės įrangos kodo kūrėjų veiklos ataskaitų teikimo įrankius, įskaitant grafinį rezultatų atvaizdavimą.
7. Turi būti galimybė kontroliuoti naudotojų prieigą prie pirminio kodo saugyklų (angl. repository). Turi būti galimybė apibrėžti naudotojo teisę matyti ar keisti pirminį kodą saugykloje.
8. Turi būti realizuota galimybė administratoriaus nurodytu el. paštu informuoti apie naują pirminio programos teksto sujungimo užklausą.
9. Turi būti realizuotas pranešimų publikavimas naudojant saityno gaudykles (angl. Webhook) apie tokius įvykius:
   1. Pirminio kodo įkėlimas (git push).
   2. Sujungimo užklausos sukūrimas (pull request).
   3. Naujos šakos (angl. Branch) ar žymos sukūrimas (angl. Tag).
10. Turi būti realizuota galimybė pirminio kodo valdymo sistemos administratoriui konfigūruoti saityno gaudykles (saityno gaudyklių adresus ir įvykių tipus).
11. Turi būti neribojamas pirminio programos teksto saugyklų skaičius.
12. Galimybė kurti, redaguoti, ištrinti, priskirti problemas (angl. issue), susijusias su pirminiu kodu saugykloje (angl. repository).
13. Turi teikti problemų (angl. issue) sekimo galimybes, įskaitant klaidų, funkcijų ir kitų problemų sekimo palaikymą.
14. Turi teikti projekto valdymo galimybes saugykloje, įskaitant eigos sekimo, užduočių priskyrimo ir terminų nustatymo palaikymą.
15. Turi turėti "wiki" ir dokumentacijos galimybes, pvz., kurti ir redaguoti dokumentaciją bei sekti pakeitimus.
16. Turi turėti techninį palaikymą iš tiekėjo.
17. Visi pirminio programos teksto saugyklos komponentai turi būti suderinami su Perkančiosios organizacijos VMWare virtualizacijos infrastruktūra, t.y. komponentai neturi reikalauti atskiros techninės įrangos ar specializuotų įrenginių. Visi komponentai turi būti diegiami VMWare virtualiose mašinose, kurios gali užtikrinti pirminio programos teksto saugyklos aukštą patikimumą.
18. Turi būti pateiktos reikalingos nuolatinio galiojimo (angl. Perpetual) licencijos arba licencijų prenumeratos (angl. Subscription).
19. Tiekėjas turi suderinti ir parengti pirminio kodo valdymo sistemos diegimo architektūrą su Perkančiąja organizacija.
20. Užbaigus diegimą tiekėjas turi suteikti 10-ties dienų stabilizavimo laikotarpį.
21. Tiekėjas turi parengti ir Perkančiajai organizacijai perduoti išsamius architektūros dokumentus.
22. Pirminio kodo valdymo sistemos architektūros kūrimo, dokumentavimo ir diegimo darbai turi būti atlikti per 12 mėn. nuo sutarties įsigaliojimo.

## Nuolatinio integravimo ir nuolatinio diegimo (CI/CD) valdymo reikalavimai

1. Turi būti atlikta aplikacijų laidų sukūrimo procesų analizė ir projektavimas.
2. Tiekėjas, įvertinęs naudojamus aplikacijų kūrimo procesus, praktikas, priemones ir būdus, ir vadovaudamasis analizės metu surinkta informacija, turi apibrėžti aplikacijų laidų sukūrimo ir diegimo procesą, apimantį veiklas nuo pirminio programos teksto pateikimo iki diegimo darbinėse aplinkose.
3. Aplikacijos laidų kūrimas ir diegimas turėtų būti atliekamas DevSecOps būdu. „DevSecOps“ turėtų būti automatizuotas naudojant CI/CD komandų grandines, kurios automatizuoja aplikacijų kūrimą nuo pirminio kodo iki diegimo darbinėse aplinkose, užtikrinant, kad saugumo ir kokybės patikra būtų integruota proceso metu.

### Reikalavimai komandų grandinėms (angl. pipelines)

1. „Application Build Automation Pipeline“ turėtų automatizuoti aplikacijos kodo kompiliavimą, testavimą ir pakavimą į konteinerių vaizdus (angl. container images) pagal šiuos reikalavimus:
   1. Automatizuoti programos kodo kompiliavimą, pakavimą (angl. build) ir testavimą, įskaitant kelių programavimo kalbų ir karkasų (angl. framework) palaikymą.
   2. Integruoti kodo kokybės skenavimą, kad butų aptiktos kodo problemos ir pažeidžiamumai, įskaitant tokių įrankių kaip „SonarQube“ palaikymą.
   3. Integruoti kodo pažeidžiamumo skenavimą, kad būtų aptikti galimi kodo saugos pažeidžiamumai, įskaitant tokių įrankių kaip OWASP palaikymą.
   4. Automatizuoti pirminio kodo pakavimą į konteinerius, įskaitant „Docker“ ir kitų konteinerių platformų palaikymą.
   5. Integruoti pirminio kodo valdymo sistemomis, tokias kaip "Git", kad būtų galimybė automatizuoti kompiliavimo, pakavimo (angl. build) ir testavimo procesą (-us).
   6. Automatizuoti pirminio kodo testavimą, įskaitant modulių testavimą (angl. unit test), integravimo testavimą ir vartotojo sąsajos testavimą.
   7. Teikti ataskaitas ir apžvalgos puslapius, kad būtų rodomi kompiliavimo, pakavimo ir testavimo rezultatai, įskaitant kodo kokybės ir saugos spragų sąrašą.
   8. Sukurti kiekvieno sukurto konteinerio atvaizdo atestavimo įrašą, kuriame būtų faktai apie sėkmingą visų tipų skenavimų ir testų, atliktų komandos grandinės (angl. pipeline) vykdymo metu, įvykdymą, įskaitant: konteinerio atvaizdo identifikatorių, skenavimo ir testų rezultatus su sėkmės/nesėkmės būsena, skenavimo ir testo vykdymo laiko žymas, komandų grandinės, kuri sukūrė atestavimo įrašą, identifikatorių.
2. „Application Promotion Automation Pipeline“ turėtų automatizuoti aplikacijų konteinerių diegimą įvairiose aplinkose, įskaitant vystymo (angl. development), testavimo (angl. test), inscenizacijos (angl. staging) ir darbinėje (angl. prod), laikantis šių reikalavimų:
   1. Automatizuoti aplikacijų konteinerių diegimą įvairiose aplinkose, įskaitant konteinerių orkestravimo platformų, tokių kaip "Kubernetes", palaikymą.
   2. Integruoti aplikacijos saugos skenavimus, įskaitant SAST, DAST ir IAST, kad būtų galima nustatyti galimus aplikacijos saugos pažeidžiamumus.
   3. Įdiegti "GitOps" būdu pagrįstą paaukštinimą, kai aplikacijos pakeitimai automatiškai įdiegiami įvairiose aplinkose, remiantis "Git" pateikimais (angl. commit) ir sujungimo (angl. pull) užklausomis.
   4. Naudoti "Git" kaip vienintelį tiesos šaltinį diegimo konfigūracijoms ir aplikacijos kodui.
   5. Įdiegti atšaukimus (angl. rollback), atvejams kai patikra po diegimo būtų nesėkminga, taip užtikrinant, kad aplikacija bus grąžinta į ankstesnę, stabilią būseną.
   6. Palaikyti kelias diegimo strategijas, įskaitant: “blue-green” diegimus, "Canary" leidimus, “rolling” naujinimus.
   7. Realiuoju laiku stebėti ir registruoti diegimo veiklą, pateikti diegimo sėkmės ir nesėkmės rodiklius.
   8. Integruoti pasirinktus CI / CD įrankius ir procesus.
3. “Test Pipeline” turėtų automatizuoti aplikacijų testavimą, pagal šiuos reikalavimus:
   1. Atlikti ištisinius (angl. end to end) testus integruotose aplinkose.
   2. Naudoti specifinius įrankius, kad patikrintų aplikacijos funkcionalumą, našumą ir saugumą.
   3. Naudoti konkrečios aplinkos testo duomenis ir ankstesnių komandų grandinių rezultatus kaip įvestį.
   4. Išvestis pateikti standartiniu formatu, kad būtų gaunamos darnios ataskaitos.
   5. Palaikyti kelis testavimo tipus, įskaitant: integravimo, vartotojo sąsajos, sistemos integravimo, regresijos, našumo ir saugos.
   6. Naudoti automatinius testavimo įrankius, aplikacijos funkcijų tikrinimui, įskaitant: automatinio testavimo karkasus, našumo tikrinimo įrankius, saugos testavimo įrankius.
   7. Identifikuoti galimas srauto sumažėjimo ir saugos grėsmes, įskaitant: našumo problemas, saugos spragas, atitikties problemas.
   8. Integruoti pasirinktus CI / CD įrankius ir procesus.
4. “Application Quarantine Pipeline” turėtų karantinuoti aplikacijas saugumo patvirtinimui ir atestavimo patikroms, prieš diegimą į darbinę aplinką, kaip nurodyta šiuose reikalavimuose:
   1. Patikrinti, ar aplikacijos atestavimo įraše, sugeneruotame naudojant “Application Build Automation Pipeline”, yra visi darbinei aplinkai privalomi įrašai.
   2. Nuskenuoti aplikacijos konteinerio atvaizdą, patikrinti ar nėra pažeidžiamumų, įskaitant: žinomus priklausomybių (angl. dependency) ir bibliotekų pažeidžiamumus, konfigūracijos pažeidžiamumus, pvz., nesaugius prievadus ar protokolus.
   3. Atmesti paaukštinimą į darbinį konteinerių vaizdų registrą, jei konteinerio vaizdo atestavimas nepatvirtintas. Nutraukti komandų grandinę (angl. pipeline) ir neleisti konteinerio vaizdo paaukštinti į darbinę (angl. production) aplinką.
   4. Inicijuoti darbinio leidimo patvirtinimo mechanizmą, jei atestavimas sėkmingas, įskaitant: sujungimo užklausos (angl. pull request) į darbinės aplinkos "GitOps" saugyklą sukūrimą, darbinės laidos (angl. production release) peržiūros ir patvirtinimo proceso inicijavimą, išsamios informacijos apie laidą (angl. release) pateikimą, įskaitant konteinerio atvaizdo identifikatorių, skenavimo ir testų rezultatus bei laiko žymas.
5. “Production Promotion Pipeline” turėtų diegti aplikacijas darbinėje aplinkoje pagal "GitOps" metodiką ir įgyvendinti atšaukimus, jei patikra po diegimo nebūtų sėkminga, laikantis šių reikalavimų:
   1. Įdiegti aplikacijas darbinėje aplinkoje naudojant "GitOps" metodiką, kai aplikacijos pakeitimai automatiškai įdiegiami į darbinę aplinką pagal "Git" pateikimus (angl. commit) ir sujungimo (angl. pull) užklausas.
6. Automatizuoti diegimo procesą, įskaitant: "Kubernetes" resursų (pvz., “deployment”, “service”, "pods") kūrimą ir atnaujinimą, aplikacijos konfigūracijos ir nustatymų atnaujinimą, reikalingų duomenų bazių migracijų ar atnaujinimų atlikimą.
   1. Atlikti patikras po diegimo, siekiant užtikrinti, kad aplikacija tinkamai ir saugiai veiktų darbinėje aplinkoje, įskaitant: aplikacijos prieinamumo ir veikimo verifikavimą, patikrinant klaidų ar išimčių egzistavimą.
   2. Įdiegti atšaukimus (angl. rollback), jei patikra po diegimo būtų nesėkminga, taip užtikrinant, kad aplikacija bus grąžinta į ankstesnę, stabilią būseną, grąžinant ją į ankstesnę, stabilią versiją.
   3. Palaikyti kelias diegimo strategijas, įskaitant: “blue-green” diegimus, "Canary" leidimus, “rolling” naujinimus.
   4. Realiuoju laiku stebėti ir registruoti diegimo veiklą, pateikti diegimo sėkmės ir nesėkmės rodiklius.

### Reikalavimai CI/CD įrankiams

1. Turi palaikyti kelias "Git" tipo saugyklas kaip aplikacijos pirminio kodo šaltinį ir galimybę pasiekti "Git" saugyklą naudojant HTTP ir SSH protokolus.
2. Turi palaikyti laikiną artefaktų (pvz., failų, diegimo paketų, konteinerių), sukurtų nuolatinio integravimo proceso metu, saugojimą.
3. Turi būti galimybė apibrėžti komandų grandinę (angl. pipeline) tekstinio failo pavidalu (pvz., YAML, JSON ar kitu formatu)
4. Turi būti realizuotas automatinio komandų grandinės konfigūravimo, aptinkant konfigūracijos failus pirminio programos kodo saugykloje, funkcionalumas.
5. Naudojant API turi būti įmanoma paskelbti komandų grandinės būseną. Pavyzdžiui, informuoti apie sėkmingą ar nepavykusį komandų grandinės vykdymą.
6. Turi būti galimybė paskirstyti komandų sekų žingsnių vykdymą per skirtingus vykdančiuosius serverius, kontroliuojant jų vykdymo eilę.
7. Naudotojo sąsaja turi būti realizuota saityno priemonėmis bei suteikti naudotojui galimybę:
   1. Inicijuoti komandų sekos vykdymą, stebėti jos būseną sekos žingsnių lygyje bei peržiūrėti vykdymo žurnalo įrašus;
   2. Matyti komandų sekos vykdymo istoriją, testavimo rezultatų istoriją.
8. Turi palaikyti rolėmis pagrįstą prieigos valdymą (RBAC), kad būtų užtikrinta, jog tik įgalioti vartotojai galėtų pasiekti ir vykdyti komandų grandines.
9. Turi būti realizuota API sąsaja arba komandinės eilutės priemonė, leidžianti automatizuoti:
   1. Komandų grandinės inicijavimą;
   2. "Webhook" parametrų keitimą;
   3. Pirminio kodo šaltinio konfigūravimą.
10. Turi būti galimybė komandų grandinių vykdymą inicijuoti reaguojant į konkrečius įvykius (pvz., kodo pakeitimus, sujungio užklausas (angl. pull request)).
11. Turi palaikyti kelias programavimo kalbas ir karkasus, įskaitant Java, Python, Node.js.
12. Turi palaikyti "Docker" ir "Kubernetes" naudojimą konteinerių vykdymui ir orkestravimui.
13. Turi būti įgyvendinta galimybė administratoriaus nurodytu el. paštu informuoti apie sėkmingai ar nesėkmingai įvykdytą komandų grandinę.
14. Turi būti neribojamas galimų komandų grandinių skaičius. Vienu metu vykdomų užduočių skaičius turi būti apribotas tik techniniais infrastruktūros ištekliais, bet ne licenciniais apribojimais.
15. Turi gebėti nuolat suderinti (angl. reconcile) faktinę programos ir infrastruktūros būseną su norima būsena, apibrėžta "Git" saugykloje. Turi palaikyti kelias aplikacijas.
16. Galimybė pateikti audito įrašų žurnalą, kad būtų galima sekti komandų grandinių, "GitOps" konfigūracijos ir diegimo istorijos pakeitimus.
17. Turi palaikyti "Canary" leidimus, leidžiančius įdiegti naujas aplikacijos versijas vartotojų pogrupiui prieš išleidžiant ją visai vartotojų bazei.
18. Turi palaikyti atšaukimus (angl. rollback), leidžiančius greitai atstatyti veikimą po nepavykusio diegimo ar konfigūracijos pakeitimo.
19. Turi palaikyti SBOM generavimą komandų grandinėse.
20. Galimybė integruotis su kodo kokybės patikros įrankiais, tokiais kaip SonarQube.
21. Visi nuolatinės integracijos ir nuolatinio diegimo valdymo komponentai turi būti suderinami su Perkančiosios organizacijos "VMWare" virtualizacijos infrastruktūra, t. y. komponentams nereikia atskiros techninės įrangos ar prietaisų. Visi komponentai turi būti įdiegti "VMWare" virtualiose mašinose, kurios gali užtikrinti aukštą virtualios mašinos prieinamumą.
22. Turi būti pateiktos reikiamos nuolatinės arba prenumerata pagrįstos licencijos.
23. Turi turėti techninį palaikymą iš gamintojo visa prenumerata pagrįstos licencijos galiojimo laikotarpį.
24. Tiekėjas turi išanalizuoti ir paruošti nuolatinės integracijos ir nuolatinio diegimo valdymo įrankių architektūrinį sprendimą ir dizainą bei paruošti šio sprendimo schemas ir dokumentaciją prieš pradedant diegimą. Architektūrinis sprendimas turi užtikrinti aukštą nuolatinio integravimo ir nuolatinio diegimo valdymo įrankių prieinamumą.
25. Tiekėjas turi teikti 10 dienų diegimo stabilizavimo laikotarpio palaikymą.

### Reikalavimai aplikacijos saugumo testavimo įrankiams

1. Turi teikti automatinę kodo analizę, kad aptiktumėte saugos spragas, klaidas ir “code smells” pirminiame kode.
2. Palaikyti kelias programavimo kalbas, įskaitant Java, C#, C++ ir kt.
3. Naudoti taisyklėmis pagrįstą analizę, kad aptiktų kodo pažeidžiamumus ir problemas su koreguojamomis taisyklėmis ir slenksčiais.
4. Integruotis su CI / CD komandų grandinėmis, kad būtų galima automatizuoti kodo analizę ir ataskaitų teikimą.
5. Turi gebėti teikti išsamias ataskaitas ir apžvalgas, kad būtų matomos kodo kokybės ir saugos metrikos.
6. Palaikyti problemų sekimą ir valdymą, bei būtų galimybė integruoti su populiariomis problemų valdymo sistemomis.
7. Teikti kodo aprėpties (angl. code coverage) analizę, kad būtų galimybė įvertinti automatinio testavimo efektyvumą.
8. Palaikyti Git šakų (angl. branch) analizę, kad būtų galimybė aptikti kodo problemas Git šakose ir sujungimo užklausose.
9. Palaikyti sujungimo užklausos analizę, kad būtų galimybė aptikti kodo pakeitimų problemas prieš jas sujungiant.
10. Aptikti saugos pažeidžiamumus, įskaitant SQL injekciją, XSS ir kt.
11. Palaikyti CWE (Common Weakness Enumeration) pažeidžiamumų aptikimą.
12. Teikti automatinį žiniatinklio aplikacijų skenavimą, kad būtų galimybė aptikti saugumo spragas.
13. Palaikyti žiniatinklio programų “crawling and spidering” testus, kad būtų galimybė nustatyti galimus pažeidžiamumus.
14. Palaikyti aktyvų žiniatinklio aplikacijų skenavimą, kad būtų aptikti pažeidžiamumai, įskaitant injekcijas, sugadintą autentikavimą ir kt.
15. Palaikyti pasyvų žiniatinklio aplikacijų skenavimą, kad būtų aptikti pažeidžiamumai, įskaitant nesaugius slapukus, nesaugius protokolus ir kt.
16. Palaikyti "JavaScript" turinčių žiniatinklio aplikacijų skenavimą.
17. Palaikyti kelių gijų skenavimą, geresniam našumui ir trumpesniam skenavimo laikui.
18. Pateikti išsamias ataskaitas ir apžvalgas, kad būtų matomos žiniatinklio aplikacijų saugos metrikos.
19. Teikti atitikties ataskaitas, kad būtų palaikomi norminiai reikalavimai, pvz., HIPAA.
20. Palaikyti OWASP pažeidžiamumų aptikimą, įskaitant injekciją, sugadintą autentikavimą ir kt.
21. Palaikyti SSL / TLS konfigūracijų skenavimą, kad būtų aptikti pažeidžiamumai ir trūkumai.

## Konteinerių registro reikalavimai

1. Turi būti įdiegta antroji "Docker Container Storage" versija ("Docker registry v2") arba lygiavertė.
2. Turi būti realizuota galimybė veikti kaip lokalus tarpinis šliuzas tarp viešų docker registry konteinerių vaizdų registrų ir lokalių Perkančiosios organizacijos naudotojų, naudotojams pateikiant konteinerių vaizdus iš lokalaus registro.
3. Turi būti užtikrintos vaizdų kaupimo talpykloje ir veidrodinio atspindžio funkcijos(angl. mirror), siekiant pagerinti našumą ir sumažinti delsą.
4. Turi palaikyti rolėmis pagrįstą prieigos valdymą (RBAC), kad būtų užtikrinta, jog tik įgaliotieji vartotojai galėtų pasiekti ir modifikuoti konteinerio atvaizdus.
5. Turi palaikyti kelių paskyrų (angl. multi-tenant) sudarymą, leidžiantį kelioms komandoms ir organizacijoms naudoti tą patį registrą su izoliuotomis saugyklomis ir prieigos valdikliais.
6. Turi palaikyti nuolatinį konteinerio vaizdo pažeidžiamumo skenavimą.
7. Turi palaikyti kelis konteinerio atvaizdo formatus, įskaitant "Docker", OCI ir kitus.
8. Turi būti įdiegta galimybė konfigūruoti atvaizdų ir (arba) žymų ištrynimą pagal sukūrimo datą arba numerį.
9. Turi palaikyti replikaciją, leidžiančią replikuoti konteinerio atvaizdus keliuose regionuose ir duomenų centruose.
10. Turi būti įdiegtos saugyklos naudojimo valdymo priemonės, pvz., kvotos ir ataskaitos.
11. Turi palaikyti integraciją su išoriniais autentikavimo paslaugų teikėjais, tokiais kaip LDAP ir OIDC.
12. Turi teikti audito įrašų žurnalą, kad būtų galima sekti konteinerio atvaizdų ir registro konfigūracijų pakeitimus.
13. Turi turėti techninį gamintojo palaikymą visa prenumerata pagrįstos licencijos galiojimo laikotarpį.
14. Turi būti pateiktos reikiamos nuolatinės arba prenumerata pagrįstos licencijos.
15. Tiekėjas iki diegimo pradžios turi išanalizuoti ir paruošti konteinerių atvaizdų registro architektūrinį sprendimą ir dizainą, taip pat paruošti šio sprendimo schemas ir dokumentaciją. Architektūrinis sprendimas turi užtikrinti aukštą konteinerių atvaizdų registro prieinamumą.
16. Tiekėjas turi teikti 10 dienų diegimo stabilizavimo laikotarpio palaikymą.

## Artefaktų registro reikalavimai

1. Turi teikti centralizuotos saugyklos paslaugas artefaktams saugoti ir valdyti, įskaitant "Maven", npm ir kitų paketų formatų palaikymą.
2. Turi palaikyti artefaktų versijavimą, leidžiantį saugoti ir valdyti kelias artefakto versijas.
3. Palaikyti artefaktų metaduomenis, įskaitant artefaktų pavadinimo, versijos ir kitos susijusios informacijos saugojimą.
4. Turi teikti paieškos ir filtravimo galimybes, kad vartotojai galėtų greitai rasti ir atsisiųsti artefaktus.
5. Turi užtikrinti prieigos kontrolę, įskaitant rolėmis pagrįstos prieigos kontrolės (RBAC) ir teisėmis (angl. permission) pagrįstos prieigos kontrolės palaikymą.
6. Turi gebėti integruotis su CI / CD komandų grandinėmis, suteikiant galimybę automatizuoti artefaktų diegimą.
7. Palaikyti artefaktų kešavimą(angl. caching) ir dubliavimą (angl. mirroring) talpykloje, kad pagerintų našumą ir sumažintų delsą.
8. Turi turėti techninį palaikymą iš gamintojo visa prenumerata pagrįstos licencijos galiojimo laikotarpį.
9. Turi būti pateiktos reikiamos nuolatinės arba prenumerata pagrįstos licencijos.
10. Tiekėjas iki diegimo pradžios turi išanalizuoti ir paruošti artefaktų registro architektūrinį sprendimą ir projektą, taip pat paruošti šio sprendimo schemas ir dokumentaciją. Architektūrinis sprendimas turi užtikrinti aukštą artefaktų registro prieinamumą.
11. Tiekėjas turi teikti 10 dienų diegimo stabilizavimo laikotarpio palaikymą.

## Automatizuoto infrastruktūros diegimo ir valdymo reikalavimai

1. Turi būti atlikta aplikacijos aplinkų paruošimo veiksmų sekos analizė ir projektavimas.
2. Tiekėjas, įvertinęs aplinkų paruošimui naudojamus procesus, praktikas, įrankius ir metodus bei remdamasis analizės metu surinkta informacija, privalo automatizuoti aplinkų paruošimo procesą, kuris apima konteinerių platformoje ir tradicinėje infrastruktūroje atliekamus darbus.
3. Automatizuotas infrastruktūros diegimas ir valdymas turėtų parengti aplinką aplikacijos veikimui, kaip nurodyta toliau aprašytuose reikalavimuose.
   1. Automatizuoti LPAR kūrimą, tinklo ir saugyklos prijungimą, AIX operacinės sistemos ir "Oracle" duomenų bazės diegimą ir konfigūravimą.
   2. Naudoti "Git" saugyklą (angl. repository) kaip aplinkos konfigūravimo ir automatizavimo procedūrų / instrukcijų / scenarijų šaltinį.
   3. Palaikyti visą aplinkos gyvavimo ciklą, įskaitant kūrimą, keitimą ir naikinimą.
   4. Automatizuoti apkrovos balansavimo priemonės konfigūravimą ir serverio vardo konfigūravimą DNS serveryje.

## Reikalavimai įrankiams

1. Siekiant palaikyti "GitOps" metodą visiems infrastruktūros komponentams, "Oracle" duomenų bazės kiekvienai aplinkai turi būti diegiamos automatiškai ir deklaratyviai. Automatizuoto infrastruktūros diegimo ir valdymo priemonės turi atitikti šiuos reikalavimus:
   1. Turi teikti informacinių technologijų infrastruktūros konfigūracijos, programinės įrangos diegimo ir tinklo valdymo automatizavimo galimybes.
   2. Turi palaikyti serverių, virtualių mašinų, debesijos paslaugų ir tinklo įrenginių automatizuoto konfigūravimo scenarijus.
   3. Turi būti galimybė diegimo scenarijų apibrėžti tekstinio failo pavidalu (pavyzdžiui, YAML, JSON ar kitokiu formatu)
   4. Scenarijai turi būti lengvai suprantami ir redaguojami naudotojams, neturintiems programavimo įgūdžių.
   5. Turi būti galimybė integruotis su populiariomis debesijos paslaugomis (AWS, Azure, Google Cloud) bei konteinerių technologijomis (Docker, Kubernetes).
   6. Turi būti galimybė integruotis su populiariomis virtualizacijos platformomis (VMWare vSphere, IBM PowerVM, IBM PowerVC)
   7. Turi turėti centralizuotą valdymo sistemą, kuri suteikia grafinę vartotojo sąsają automatizavimo procesų valdymui.
   8. Turi nenaudoti agentų ir prie valdomų sistemų jungtis per SSH arba API.
   9. Grafinė vartotojo sąsaja turi turėti rolėmis pagrįstą prieigos valdymą ir palaikyti išorinius autentikavimo teikėjus, tokius kaip LDAP ir OIDC.
   10. Turi būti galimybė stebėti informacinių technologijų inventorių ir valdyti užduočių vykdymą pasirinktoms informacinių technologijų inventoriaus grupėms.
   11. Turi būti galimybė praplėsti automatizavimo galimybes integruojant papildomus modulius ir pritaikant juos pagal poreikius.
   12. Turi būti įmanoma vykdyti užduotis / veiksmus, kuriuos sukelia įvykiai (angl. events) arba žiniatinklio gaudyklės (angl. webhooks).
   13. Turi turėti API, leidžiančią keisti užduočių / veiksmų parametrus, žiniatinklio gaudykles, valdomą infrastruktūrą.
   14. Turi palaikyti "infrastruktūros kaip kodo" principus, deklaratyvius norimos infrastruktūros būklės aprašymus.
   15. Turi turėti techninį palaikymą iš gamintojo.
2. Turi būti suteikta prieiga prie gamintojo dokumentų.
3. Turi būti pateiktos reikiamos nuolatinės arba prenumerata pagrįstos licencijos leidžiančios valdyti bent 100 taškų.
4. Tiekėjas privalo išanalizuoti ir parengti automatizuoto infrastruktūros diegimo ir valdymo sistemos architektūrinį sprendimą ir dizainą, bei suderinti jį su Perkančiąja organizacija.
5. Tiekėjas turi teikti 10 dienų diegimo stabilizavimo laikotarpio palaikymą.
6. Tiekėjas parengia ir Perkančiajai organizacijai pateikia detalius architektūrinius dokumentus.

## Platformos Priežiūros paslaugų reikalavimai

1. Tiekėjas turi turėti Perkančiosios organizacijos darbo dienomis darbo valandomis veikiančią pagalbos tarnybą.
2. Tiekėjas turi turėti pagalbos tarnybą komunikuojančią lietuvių kalba su Perkančiąja organizacija raštu ir žodžiu.
3. Tiekėjo pagalbos tarnyba turi suteikti galimybes registruoti kreipinius įvairiais nurodytais kanalais: elektroniniu paštu; fiksuoto ir mobilaus ryšio telefonu; naudojant WEB sąsają.
4. Tiekėjas turi būti įdiegęs veikiančius ir aprašytus incidentų bei keitimų valdymo procesus, atitinkančius informacinių technologijų paslaugų valdymo (ITIL ar lygiavertės metodikos) gerųjų praktikų rekomendacijas bei veikiantį internetinį portalą kreipiniams registruoti bei peržiūrėti.
5. Tiekėjo pagalbos tarnyba turi užtikrinti operatyvų atgalinį ryšį ir informacijos apie incidentus realiu laiku (angl. On-line) teikimą interneto tinklalapyje, veikiančiame HTTPS protokolu.
6. Pagalbos tarnyba turi informuoti apie užregistruotų incidentų būklę, planuojamą incidentų išsprendimo datą ir laiką bei incidentų išsprendimą.
7. Tiekėjas turi būti įsidiegęs sprendimą, kurio pagalba visi nuotoliniai prisijungimai būtų įrašomi, o esant Perkančiosios organizacijos poreikiui, Tiekėjas turi suteikti Perkančiajai organizacijai įrašytą sesiją. Sesijos turi būti saugomos ne trumpiau kaip 6 mėn.
8. Tiekėjas turi teikti priežiūros paslaugas platformai 5 metus.
9. Priežiūras paslaugos teikimo laikas: Darbo dienomis nuo 8:00 iki 17:00;
10. Reakcijos į visų tipų incidentus laikas: ne ilgiau kaip 2 (dvi) valandos;
11. Incidentų išsprendimo laikas: ne ilgiau kaip 8 (aštuonios) valandos;
12. Reakcijos į užklausas ir keitimus laikas: ne ilgiau kaip 4 (keturios) valandos;
13. Užklausų ir keitimų išsprendimo laikas: ne ilgiau kaip 16 (šešiolika) valandų (sudėtingesnių užklausų ir keitimų išsprendimo laikas Šalių abipusiu susitarimu gali būti keičiamas).
14. Iki kiekvieno einamojo mėnesio 10 (dešimtos) dienos Tiekėjas pateikia paslaugų kokybės parametrų ataskaitą už praėjusį mėnesį. Ataskaitoje nurodomi šie duomenys: Vidutinės incidentų reakcijos bei išsprendimo trukmės, visų incidentų sąrašas su nurodytais reakcijos bei išsprendimo laikais.
15. Turi būti užtikrinta visos techninės ir programinės įrangos stebėsena (365, 24x7) bei turi būti siunčiami automatiniai pranešimai apie incidentus.
16. Tiekėjas turi pateikti ir naudoti stebėsenos sistemą, skirtą stebėti visai Perkančiosios organizacijos informacinių technologijų infrastuktūros elementams, kurie paminėti techninė specifikacijoje;
17. Stebimi parametrai turi būti suderinti su Perkančiąja organizacija;
18. Esant poreikiui, Tiekėjas Perkančiosios organizacijos prašymu turi naujus stebėjimų elementus įtraukti į stebėsenos sistemą;
19. Tiekėjas turi turėti apsirašęs slaptažodžių valdymo procesą. Visi slaptažodžiai gauti iš Perkančiosios organizacijos turi būti saugomi specializuotame įrankyje, kuris turi šifruoti slaptažodžius AES-256 šriftu arba lygiaverčiu.
20. Prisijungimas prie įrankio turi būti bent dviejų faktorių.
21. Įrankis turi kaupti visus slaptažodžių panaudojimo žurnalinius įrašus.
22. Prisijungimas prie Perkančiosios organizacijos tarnybinės stoties (angl., server) nuotoliniu būdu turi būti atliekamas specializuotos programinės įrangos pagalba, kurioje realizuota dviejų lygių autentifikacija, užtikrinant, kad tik patvirtinti specialistai galės atlikti prisijungimą prie tarnybinės stoties. Visi prisijungimai ir jų trukmės turi būti fiksuojamos specializuotoje programinėje įrangoje. Prisijungimo prie tarnybinės stoties kanalas turi būti šifruojamas specializuotos programinės įrangos pagalba.
23. Tiekėjas turi būti įsidiegęs sprendimą, kurio pagalba visi nuotoliniai prisijungimai ir jų laikai būtų įrašomi, ir esant Perkančiosios organizacijos poreikiui, suteikti įrašytą sesiją.
24. Sesijos įrašai turi būti saugomos ne trumpiau kaip 6 mėn.
25. Nuotolinės prieigos prie serverių kanalas turi būti užšifruotas naudojant specializuotą programinę įrangą.
26. Tiekėjas turi spręsti platformos incidentus pagal reakcijos ir sprendimo laikus;
27. Tiekėjas turi atlikti platformos pakeitimus, kas susiję su esamos platformos konfigūracijos pakeitimais;
28. Tiekėjas turi pro aktyviai reguliariai atnaujinti platformos komponentus;
29. Priežiūros paslaugas Tiekėjas turi teikti pagal gerąsias ITIL praktikas.
30. Aplinkosauginiai reikalavimai:

206.1. Atsižvelgiant į tai, kad perkamos Paslaugos yra nematerialaus pobūdžio intelektinės paslaugos, nesusijusios su materialaus objekto sukūrimu, kurių teikimo metu nebus sukurtas neigiamas poveikis aplinkai, taip pat nebus sukuriamas taršos šaltinis ar generuojamos atliekos, vadovaujantis Aplinkos apsaugos kriterijų taikymo, vykdant žaliuosius pirkimus, tvarkos aprašo, patvirtinto Lietuvos Respublikos aplinkos ministro 2011 m. birželio 28 d. įsakymu Nr. D1-508, Pirkimas laikomas žaliuoju.