**SIA "Sustainable Mobility Technologies"**
Rīga, Pirts iela 9 – 3
Rīga, LV-1003
Latvija

**Iepirkuma SMT-IEP/VMKC-02**
**"Programmēšanas pakalpojumi Elektrisko skrejriteņu infrastruktūras vadības IT sistēmas izstrādei"**

**TEHNISKĀ SPECIFIKĀCIJA**

*Apstiprināts ar SIA "Susainable Mobility Technologies" iepirkumu komisijas 2025. gada 2. jūlija lēmumu Nr. IK/SMT-IEP/VMKC-02-1*

Pakalpojuma galvenais mērķis ir Pasūtītāja uzdevumā izveidot centralizētu programmatūras risinājumu, kas ļauj dažādu operatoru elektriskajiem skrejriteņiem droši un autonomi uzlādēties uz pilsētā izvietotām bezvadu stacijām. Sistēma koordinēs uzlādes sesijas, lietotāju finanšu stimulus un savienojumus ar operatoru esošajām flotes pārvaldības sistēmām, kā arī nodrošinās datu apmaiņu ar uzlādes iekārtu malējās skaitļošanas vienībām.

Pētījums tiek veikts projekta ""Videi draudzīgas skrejriteņu fotoelektriskās (PV) uzlādes infrastruktūras vadības sistēmas izstrāde"" aktivitāšu ietvaros. Projekta Nr. 2.2.1.3.i.0/1/24/A/CFLA/004.

# I Daļa: Darba uzdevums

**Funkcionālās prasības**

Sistēmai jāidentificē katrs skrejritenis, izmantojot unikālu aparatūras ID, un jāuzsāk sesijas ieraksts vienas sekundes laikā pēc fiziskas novietošanas uz uzlādes paneļa.

Risinājumam jāuzkrāj sesiju žurnāli, depozīta transakcijas un operatoru konfigurācijas, nodrošinot ACID īpašības un iespēju meklēt jebkuru ierakstu piecu sekundžu laikā.

Lietotāju stimulu modulim jāatbalsta depozīta atmaksa un tieši finanšu pārskaitījumi uz norādītajiem kontiem saskaņā ar ISO 20022 maksājumu standartiem.

Operatoru vadības panelī jābūt reāllaika kartei, statistikas pārskatiem un API atslēgu pārvaldībai.

Visām ārējām integrācijām jāizmanto REST (OpenAPI 3.1) un WebSocket straumes ar vienotu OAuth 2.1 autorizācijas kārtību.

Finansē Eiropas Savienība NextGenerationEU
2027 Nacionālais attīstības plāns

## 2. Nefunkcionālās prasības

Veiktspēja: sistēmai jāapkalpo vismaz 5 000 vienlaicīgu pieslēgumu un jāspēj rakstīt 100 uzlādes ierakstus sekundē ar 95 percentiļu atbildes laiku, kas nepārsniedz 300 milisekundes.

Mērogojamība: risinājumam jābūt konteinerizētam, izmantojot Docker, un izvietojamam Kubernetes vai alternatīvā orkestrēšanas vidē ar horizontālās paplašināšanas atbalstu.

Drošība: dati transporta plūsmā jāšifrē ar TLS 1.3, atmiņā un diskā – ar AES-256. Risinājumam jāatbilst ISO/IEC 27001, OWASP Top 10 un GDPR.

Pieejamība: jānodrošina 99.9 % uptime mēnesī, izmantojot vismaz divu datu centru redundanci.

Lietojamība: lietotāja interfeisi jāveido saskaņā ar WCAG 2.2 AA.

## 3. Tehniskie nosacījumi un arhitektūra

Izstrādājamā IT sistēma tiks balstīta uz daudzslāņu mikroservisu arhitektūras principiem, kas nodrošina sistēmas modularitāti, augstu pieejamību, uzticamību un mērogojamību. Katrā loģiskajā slānī (prezentācijas, lietojumprogrammu, datu) darbosies neatkarīgi servisi, kuri mijiedarbosies caur skaidri definētiem API un notikumu apmaiņas mehānismiem.

Sistēmas kodols atradīsies mākoņa vidē, izmantojot Kubernetes konteineru orķestrēšanas risinājumu, kas nodrošina elastīu un augstu pieejamību izvietošanā. Lai nodrošinātu reāllaika datu apstrādi un starpservisu komunikāciju, paredzēta Apache Kafka vai RabbitMQ Streams ieviešana ar notikumu virzītas arhitektūras atbalstu.

Malējās skaitļošanas ierīcēs darbosies gRPC pamatots komunikācijas aģents, kas uztur savienojumu ar centrālo sistēmu, nodrošinot zemu latentumu un uzticamu datu nodošanu. Datu uzglabāšanai tiks izmantota PostgreSQL datubāze ar TimescaleDB paplašinājumu laika rindas datiem, savukārt operatīvai kešošanai tiks pielietots Redis.

Drošības slānī sistēma ieviesīs TLS 1.3 protokolu komunikācijai starp komponentiem, AES-256 standartu datu šifrēšanai atmiņā un diskā, un OAuth2 protokolu lietotāju autentifikācijai, papildinot to ar JSON Web Token (JWT) mehānismu. Lomas un piekļuves tiesības tiks pārvaldītas ar RBAC (Role-Based Access Control) modeli.

Visi sistēmas komponenti nodrošinās observabilitāti, izmantojot OpenTelemetry, kas ķauj centralizēti apkopot metriku, izsekojamību un logus. Datu un servisu stāvokļu vizualizācija tiks nodrošināta ar Prometheus un Grafana. Šāda arhitektūra nodrošina platformas gatavību ražošanas ieviešanai un turpmākai paplašināšanai.

# II Daļa: Darba izpilde

**1. Sistēmanalīze un vajadzību kartēšana**

Pirmajā posmā plānots veikt padziļinātu sistēmanalīzi, kuras mērķis ir identificēt visus funkcionālos, tehniskos un lietošanas prasību aspektus. Darbs ietver strukturētas intervijas ar skrejriteņu operatoriem, pilsētas infrastruktūras pārvaldītājiem un gala lietotājiem, lai apzinātu esošos procesus, problēmpunktus un integrācijas vajadzības.

Rezultātā tiks izstrādāts prasību dokuments, kurā būs definētas gan funkcionālās, gan nefunkcionālās prasības, drošības aspekti, servisa līmeņa indikatori (SLA) un sistēmas robežu nosacījumi. Šis dokuments kalpos kā pamatmateriāls turpmākajā arhitektūras izstrādes posmā.

Plānotais apjoms: 160 stundas

**2. Sistēmas arhitektūras projektēšana**

Balstoties uz iepriekšējā posmā iegūtajiem secinājumiem, tiks izstrādāta detalizēta sistēmas arhitektūra. Tā ietvers komponentu sadalījumu mākoņa un malējās skaitļošanas līmeņos, datubāžu shēmas, API struktūru definīciju (ar pilnu specifikāciju atbilstoši OpenAPI 3.1), drošības risinājumu aprakstu (TLS 1.3, OAuth2, JWT, RBAC) un datu apmaiņas mehānismus (REST, WebSocket).

Arhitektūras dokuments būs centrālais pamats visiem nākamajiem izstrādes posmiem, nodrošinot loģisku un savstarpēji savietojamu komponentu attīstību.

Plānotais darba apjoms: 80 stundas

**3. Sistēmas pamatfunkcionalitātes izstrāde**

Šajā posmā programmatūras inženieri izstrādās pirmos funkcionālos komponentus, balstoties uz iepriekšējās arhitektūras specifikācijas.

Tiks izveidots modulis, kas nodrošina skrejriteņu identifikāciju uzlādes brīdī, izmantojot unikālu ID no malējās ierīces.

Paralēli tiks izveidota datu glabāšanas infrastruktūra (PostgreSQL/TimescaleDB), kas uzkrās uzlādes sesiju žurnālus un lietotāja iesaistes informāciju depozīta sistēmā.

Tiks implementēta lietotāja autentifikācija un autorizācija, kā arī operatoru vadības saskarne. Visas izstrādes darbības tiks veiktas uz versiju kontroles un CI/CD infrastruktūras pamata.

Rezultāts būs pilnīgi funkcionala kodu bāze ar testēšanas vides atbalstu, kas būs pamats alfa versijas izstrādei.

Plānotais apjoms: 480 stundas

**4. Eksperimentālā modeļa (alfa versijas) izstrāde**

Pamatojoties uz pamatfunkcionalitātes izstrādi, tiks veikta sistēmas integrācija un attīstība TRL5 līmenī. Tiks ieviesti REST un WebSocket servisi datu apmaiņai ar malējām ierīcēm, kā arī lietotāja saskarne, kas nodrošina depozīta funkcionalitāti.

Alfa versija tiks testēta reālos apstākļos, veicot veiktspējas mērījumus un vienībtestus. Testēšanas rezultāti tiks izmantoti, lai identificētu sistēmas stabilitātes un mērogošanas riskus, kuri tiks novērsti nākamajā posmā.

Alfa versijas kvalitāte noteiks bāzi beta funkciju attīstībai.

Plānotais apjoms: 960 stundas

**5. Beta versijas izstrāde un testēšana (TRL6)**

Noslēdzošajā posmā, balstoties uz alfa versijas testēšanas rezultātiem, tiks veikta koda un sistēmas optimizācija. Datubāzes darbības efektivitāte tiks uzlabota, UI pilnveidots atbilstoši operatoru vajadzībām, un tiks integrēti eksperimentālā AI moduļi, kas, piemēram, piedāvās lietotājiem maršruta suģestēšanu, balstoties uz pieejamajām uzlādes kapacitātēm.

Tiks veikti slodzes testi, simulējot vairākus tīkstošus vienlaicīgo lietotāju, un lietotāju akceptēšanas testēšana.

Šīs fāzes rezultātā taps pilnīgi funkcionāla, testēta un dokumentēta sistēma, kas būs gatava integrācijai ar pilotprojekta stacijām reālā pilsētas vidē.

Plānotais apjoms: 480 stundas

**6. Nodevumi**

Katra projekta posma noslēgumā paredzēts nodot skaidri definētus rezultātus, kas apliecina izstrādes progresu un nodrošina bāzi nākamajai izstrādes fāzei.

1. **Sistēmanalīzes dokuments**: satur lietotāju vajadzību aprakstu, esošo procesu analīzi, funkciju sarakstu, nefunkcionālās prasības, drošības apsvērumus un sistēmas integrācijas karti. Šis dokuments kalpos kā primārā atsauce arhitektūras izstrādē.
2. **Arhitektūras apraksts**: ietver vispārējo sistēmas komponentu shēmu, datubāzes ER diagrammas, servisu un API savienojumu karti, drošības modeļu dokumentu un konfigurācijas vadlīnijas.
3. **Funkcionāls prototips**: satur programmatūras pirmkodu, konfigurētu testēšanas vidi, CI/CD pipeline aprakstu, galveno komponentu testu komplektu, lietotāja autentifikācijas un operatora paneļa prototipu.

4. **Alfa versija**: satur integrētu sistēmu ar malējo ierīču atbalstu, lietotāju motivācijas saskarni, REST un WebSocket API, vienībtestiem un veiktspējas atskaiti. Šīs versijas testēšanas rezultāti tiks izmantoti beta versijas pilnveidošanai.
5. **Beta versija**: satur sistēmas pilnu funkcionalitāti ar AI komponentēm, veiktspējas un slodzes testu rezultātiem, akceptēšanas testu dokumentāciju, lietotāju rokasgrāmatu un galīgo izstrādes dokumentāciju, kas nodrošina gatavību sistēmas ieviešanai ekspluatācijā.

# III Daļa Darba organizācija un kontrole

Projekta izstrāde tiks veikta iteratīvā un inkrementālā pieejā, balstoties uz Scrum metodoloģiju, kas ietver regulārus divu nedēļu sprintus. Katra sprinta ietvaros tiks veikta konkrētu funkcionalitātes vienību izstrāde, testēšana un nodošana demonstrācijai pasūtītājam. Projektā tiks definēts produkta īpašnieks no pasūtītāja puses, kas nodrošinās savlaicīgu prasību precizēšanu un akceptēšanu.

**Starpposmu nodevumi un pārbaudes**

Izstrādes komanda darbosies ar versiju kontroli (Git), un CI/CD mehānisma ietvaros (piemēram, GitLab CI vai GitHub Actions) tiks nodrošināta nepārtraukta integrācija un izvietošana. Katrs jauns funkcionalitātes piegādes cikls tiks papildināts ar vienībtestiem, integrācijas testiem un drošības validācijām.

Dokumentācija tiks veidota paralēli izstrādei, nodrošinot, ka katram komponentam, API galapunktam un datu struktūrai ir apraksts, kas pieejams gan izstrādātājiem, gan pasūtītāja pārstāvjiem. Projekta gaitā regulāri notiks sprinta plānošanas, statusa sanāksmes un retrospektīvas, lai nodrošinātu caurspīdīgu progresu un savlaicīu problēmu risināšanu.