

**VENTSPILS AUGSTSKOLA**  
**INFORMĀCIJAS TEHNOLOGIJU FAKULTĀTE**

**BAKALAURA DARBS**

**SENSORU UN BEZPILOTA GAISA KUGA DATU  
INTEGRĀCIJA UN VIZUALIZĀCIJA GEOGRĀFISKAJĀS  
INFORMĀCIJAS SISTĒMĀS**

Autors

Ventspils Augstskolas

Informācijas tehnoloģiju fakultātes

bakalaura studiju programmas  
„Datorzinātnes”

3. kurga students

**Jēkabs Konošonoks**

Matr.nr. 22020030

---

(paraksts)

Fakultātes dekāns

doc. Dr.sc.comp. Vairis Caune

---

(paraksts)

Zinātniskā vadītāja

Dr.sc.ing. Ginta Majore

---

(paraksts)

Recenzents

---

(ieņemamais amats, zinātniskais nosaukums, vārds, uzvārds)

---

(paraksts)

Ventspils  
2025

# ANOTĀCIJA

**Darba nosaukums:** Sensoru un bezpilota gaisa kuģa datu integrācija un vizualizācija ģeogrāfiskajās informācijas sistēmās.

**Darba autors:** Jēkabs Konošonoks

**Darba vadītāja:** Dr.sc.ing. Ginta Majore

**Darba apjoms:** 63 lpp., 1 tabulas, 31 attēli, 5 pielikumi.

**Atslēgas vārdi:** STREAMLIT, GIS, WEBODM, ORTOFOTO.

Bakalaura darba mērķis ir izstrādāt informācijas sistēmas prototipu, kurā būtu integrēti sensoru un bezpilota gaisa kuģa (dronu) dati vienotā laika sērijā ģeogrāfiskās sistēmas ietvarā.

Tīmekļa vietne tiek izstrādāta, izmantojot Streamlit – Python tīmekļa vietņu izstrādes ietvaru, kā arī Folium kartēšanas pakotni ortofoto un sensoru datu vizualizācijai un WebODM ortofoto un VARI apstrādei. Lietotāju un sensoru ierīces koordinātu ilgtermiņa uzglabāšanu nodrošina PostgreSQL datubāze.

Tīmekļa vietne ir paredzēta lauksaimniekiem, lai atvieglotu lauka augu vizuālo analīzi, izmantojot lauka ortofoto un augu zaļuma indeksa (VARI) attēlus, kā arī integrējot temperatūras un mitruma sensora datus kā pārklājuma slāņa zonas kartes skatā.

## ANNOTATION

**Title:** IoT and UAV data integration and visualization within GIS.

**Author:** Jēkabs Konošonoks

**Academic Advisor:** Dr.sc.ing. Ginta Majore

**Volume of the work:** 63 pages, 1 tables, 31 images, 5 attachments.

**Keywords:** STREAMLIT, GIS, WEBODM, ORTHOPHOTO.

The aim of the bachelor's thesis is to develop a prototype of an information system that would integrate sensor and unmanned aerial vehicle (drone) data into a single time series within a geographic system.

Development of the website is done using Streamlit, a python web development framework, as well as the Folium mapping package for orthophoto and sensor data visualization and WebODM for orthophoto and VARI data processing. Long-term storage of user and sensor device coordinates is implemented by a PostgreSQL database.

The website is designed for farmers to facilitate visual analysis of field plants using field orthophotos and plant greenness index (VARI) images, as well as integrating temperature and humidity sensor data as overlay areas in a map view.

# SATURS

|  |    |
|--|----|
| SAĪSINĀJUMU UN NOSACĪTO APZĪMĒJUMU SARAKSTS.....           | 6  |
| IEVADS.....  | 7  |
| 1. LĪDZĪGU RISINĀJUMU SALĪDZINĀJUMS .....                  | 10 |
| 1.1. PIX4Dfields risinājums.....                           | 10 |
| 1.2. Agremo risinājums.....                                | 11 |
| 1.3. QGIS risinājums .....                                 | 12 |
| 2. SISTĒMAS PRASĪBAS .....                                 | 14 |
| 2.1. Eksistējošo ievada datu apraksts .....                | 14 |
| 2.2. Funkcionālās prasības .....                           | 15 |
| 2.3. Nefunkcionālās prasības .....                         | 17 |
| 3. TEHNOLOGISKIE RISINĀJUMI UN TO SALĪDZINĀJUMS .....      | 19 |
| 3.1. Priekšpuses tehnoloģijas .....                        | 19 |
| 3.2. Aizmugursistēmas tehnoloģijas .....                   | 21 |
| 4. SISTĒMAS IZSTRĀDE UN TESTĒŠANA .....                    | 25 |
| 4.1. Datubāzes sistēma.....                                | 25 |
| 4.2. WebODM API savienojums .....                          | 27 |
| 4.3. Autorizācijas sistēma .....                           | 29 |
| 4.4. Ortofoto izveides sistēma.....                        | 30 |
| 4.5. Lietotāju ortofoto kartes pārvaldes sistēma .....     | 34 |
| 4.6. Sensoru datu iestatīšana .....                        | 35 |
| 4.7. Sensoru datu un ortofoto vizualizēšana GIS vidē ..... | 36 |
| 4.8. Sensoru datu diagramma.....                           | 40 |
| 4.9. Sistēmas prototipa testēšana .....                    | 41 |
| 5. SISTĒMAS DARBĪBAS APRAKSTS UN PĀRSKATS .....            | 44 |
| 5.1. Lietotāja interfeisa apraksts .....                   | 44 |
| 5.2. Sistēmas veikspējas un uzturēšanas aspekti .....      | 52 |
| SECINĀJUMI UN PRIEKŠLIKUMI.....                            | 53 |

|  |    |
|--|----|
| IZMANTOTĀS LITERATŪRA UN AVOTU SARKSTS ..... | 56 |
| PIELIKUMI .....                              | 58 |
| GALVOJUMS .....                              | 63 |

## **SAĪSINĀJUMU UN NOSACĪTO APZĪMĒJUMU SARAKSTS**

API – Lietojumprogrammas saskarne

GeoTIFF – TIFF faila metadatu standarts, kas glabā ģeogrāfiskās atsauces

GIS – Ģeogrāfiskā informācijas sistēma

JSON – Strukturēts datu apmaiņas formāts

CSS – Stila lapas kaskadēšana

WebODM – Atvērtā pirmkoda rīkkopa un API aerofoto attēlu apstrādei

JWT – Interneta standarts datu izveidei, kuru vērtums satur JSON, kas apgalvo noteiktu skaitu prasību

VARI – Redzamais atmosfēras izturības indekss

## **IEVADS**

Lauksaimniecība ir viena no vissvarīgākajām industrijām Latvijā. Tā nodrošina tūkstošiem lauksaimniekiem iztiku, kā arī uzturu valsts iedzīvotājiem. Lauksaimniecība un tās raža ir civilizācijas pamats, jo bez tās cilvēki nespēj dzīvot. Šī nozare ir bijusi centrālā un galvenā profesija Latvijā jau simtiem gadu, un autors uzskata, ka tās ietekme ir skaidri redzama mūsdienās latviešu kultūrā un psihē.

Ar 21. gadsimta strauji augošo tehnoloģiju, visu mērogu lauksaimniekiem, it īpaši mazāka izmēra neatkarīgiem lauksaimniekiem ir grūti tikt līdzi šai augsti pieprasītai industrijai, kura nepārtraukti attīstās. Galvenie izaicinājumi, kuri lauksaimniekiem jārisina ir: ražas izsekošana visās augšanas fāzēs, ātra auga slimības un labklājības atpazīšana, vides faktoru: gaisa un augsnes mitruma un temperatūras izsekošana ilgā laika periodā, kā arī vizuāla ražas analīze, lai noteiktu neveselīgus ražas segmentus un pieņemt lēmumu, vai miglot lauku, vai izsmidzināt barības vielas utt. Šādus izaicinājumus mēģina risināt darba zinātniskās vadītājas uzņēmums, sadarbībā ar kuru tiek veidots šis darbs.

Pēc autora domām, vienīgais ceļš uz priekšu ir lauksaimniekiem attīstīties kopā ar šīm jauni veidotām un revolucionārām tehnoloģijām – ortofotogrāfiju, bezpilota gaisa kuģiem, mitrumu un temperatūras sensoriem, integrējot šīs tehnoloģijas kopā vienā viegli lietojamā risinājumā, kas palīdzēs visu mērogu lauksaimniekiem palielināt ražas apjomu, atvieglo lēmumu pieņemšanu un palīdzēt atrast un noteikt auga labklājības ietekmes faktorus. Bet darba autora esošo risinājumu ieskats parāda, ka pašlaik neeksistē rīks vai programmatūra, kas spēj paveikt visas nepieciešamas sistēmas prasības vienā viegli lietojamā risinājumā.

Tāpēc šo darba mērķis ir izstrādāt informācijas sistēmas prototipu, kurā būtu integrēti sensoru un bezpilota gaisa kuģa dati vienotā laika sērijā ģeogrāfiskās sistēmas ietvarā. Rezultējošais prototips nodrošinās lietotājam spēju augšupielādēt bezpilota gaisa kuģu attēlus, no kuriem tiks ģenerēts pilns lauka kartes skats kopā ar integrētiem mitruma un temperatūras sensora datu informācijas pārklājumu slānjiem vienotā laika sērijā. Lai realizētu šo mērķi, būs nepieciešams veikt vairākus darba uzdevumus:

Tiks izpētīti alternatīvi risinājumi, kuri tiek piedāvāti pašreizējā lauksaimniecības programmatūras klāstā. Autors strukturēti atbildēs uz jautājumiem: kādas ir piedāvātās funkcionalitātes, kam risinājums ir domāts, vai risinājums ir maksas, vai tiek atbalstītas darba mērķa pamata prasības: bezpilota gaisa kuģa ortofoto bildes sašūšana un sensora datu integrācija un vizualizāciju GIS vidē.

Izpētīsim visus populāros un atbilstošos tehnoloģiskos risinājumus un noskaidrosim, kuri tuvāk atbilst darba mērķa realizācijai.

Tiks aprakstītas vispiemērotākās Python bibliotēkas, kas visakurātāk atbildēs prototipa funkcionalajām prasībām kā programmas vide (tūmekļa vai lokāla aplikācija), datu uzglabāšana, kartes un sensora datu vizualizāciju, autentifikāciju, sesijas pārvalde, datu integrācijas atbalsts, lietošanas vienkāršība. Kā arī tiks atrasta vispiemērotākā datu glabāšanas arhitektūra un bezpilota gaisa kuģa attēlu sašūšanas operācijas API, ko prototips izmanto, lai nodrošinātu lauka vizuālo funkcionalitātes izpildi.

Izstrādāsim sistēmas prototipa koncepciju, kura demonstrēs prototipa funkcionalās prasības, sistēmas arhitektūru, pamata funkcionalitāti, lietotāju atļaujas, aplikācijas darba plūsmu, sistēmas komponenšu struktūru un mijiedarbību starp komponentēm un datubāžu struktūru. Prototips tika izstrādāts pirms pilnas koncepcijas izveides, lai pārliecinātos par risinājuma iespējamību. Tāpēc koncepcija tiek integrēta prototipa prasībās un izstrādes aprakstā.

Sekojoj sistēmas funkcionalām prasībām tiks izstrādāts sistēmas prototips atsevišķos intervālos, ieviešot vienu funkcionalitāti pēc otras. Koncentrējoties pirmkārt uz fundamentālākajām prasībām, vēlāk papildinot prototipu ar sekundāram funkcionalitātēm.

Pirmajās izstrādes stadijās tiks implementēts lokāls prototips, kas dos ieskatu sistēmas pamata uzbūvē un funkcionalitātē – bezpilota gaisa kuģa bildes sašūšanā un vizualizēšanā ģeogrāfiskā informācijas sistēmā un sensora datu integrēšana jauktā risinājumā kopā ar bezpilota gaisa kuģa attēlu, kā arī atsevišķi, viegli pārskatāmās diagrammās, kas palīdzēs noteikt iespējamus uzlabojumus un izmaiņas.

Pēc tam tiks ieviesti datubāzes un mākoņa risinājumi – ortofoto koordinātu un datumu uzglabāšana, transformējot aplikāciju ar mākoņa krātuves atbalstu.

Katrā prototipa izstrādes posmā tiks veikti strukturēti manuāli sistēmas funkcionalitātes testi, nodrošinot konstantu un drošu sistēmas darbību. Veiksim sistēmas strukturētu dokumentāciju, pierakstot pilnu sistēmas darbības opcijas, plūsmu un skaidru funkcionalitātes pārskatu. Tiks aprakstīts prototipa sistēmas komponenšu interaktivitāte un pilns sistēmas lietošanas gadījums ar piemēriem, ekrānuzņēmumiem un skaidrojumiem.

Eksistē vairāki rīki, mājaslapas un lietojumprogrammas, kas spēj piedāvā atsevišķas daļas no darba mērķa nepieciešamās funkcionalitātes, bet pilnā sensoru un bezpilota gaisa kuģa bildes datu integrēšanas funkcionalitāte GIS vidē nav pieejama nevienā viegli lietojamā risinājumā. Tāpēc izveidosim jaunu risinājumu specifiski darba mērķa problēmas kopumu risināšanai.

Pētījuma mērķa realizācijā tiks veidota tīmekļa lietotne, kuras priekšpusē, kur lietotājs mijiedarbosies ar aplikāciju, tiks izmantota Python programmēšanas valoda, precīzāk, Streamlit – atvērtā pirmkoda Python programmēšanas valodas ietvars datu zinātniekiem un mašīnmācīšanās inženieriem, lai nodrošinātu dinamiskas datu lietotnes tikai ar dažām koda rindām [1]. Streamlit satvars specializējas datu apstrādē un vizualizēšanā, kā arī ir daudzas noderīgas iebūvētas funkcionalitātes, kas palīdz ar optimizētu datu apstrādi kā, piemēram, *st.cache\_data* un *st.cache\_resource*, kas saglabā funkcijas rezultātu kešatmiņā un veic jaunu funkcijas izsaukumu tikai tad, kad mainās funkcijas atkarīgie parametri.

Streamlit satvarā tiks izmantota Folium Python pakotne, kas nodrošina GIS funkcionalitāti: ortofoto vizualizēšanu, sensora ierīču koordinātas uzstādīšanu un vairāku sensoru datu slāņu vizualizēšanu. Šo funkcionalitāti nodrošina atbilstošas *FeatureGroup* kategorijas, kuras ir Folium objektu grupējumi, kuros tiks izveidoti *Circle* objekti virs sensora koordinātām, vizualizējot gaisa un augsnes temperatūru un mitru kā krāsu specifisku apla pārklājumu attiecīgi datu vērtībai. GeoTIFF faila apstrādei priekš ortofoto vizualizāciju Folium kartē – GeoTIFF koordinātas centra un robežas aprēķināšanai tiek izmantotas *Rasterio* un *pyproj* pakotnes, kā arī *NumPy* un *PIL/Pillow* pakotnes GeoTIFF attēla pikselu operācijām attēlu tukšo datu noņemšanu vajadzībām. Datubāzes nolūkiem tiks izmantots atvērtā koda relāciju datubāzes pārvaldības sistēma – PostgreSQL dēļ tās uzticamības, ātro veikspēju, datu integritāti un plašo funkcijas kopumu [2].

# **1. LĪDZĪGU RISINĀJUMU SALĪDZINĀJUMS**

Lai noskaidrotu vai autora piedāvātais rīks ir aktuāls, ir nepieciešams apskatīt un salīdzināt jau eksistējošos risinājumu no vairākiem skata punktiem: funkcionalitātes, lietojamības un pieejamības. Lauksaimniecības industrijā ir pieejami vairāki satelīta un bezpilota gaisa kuģa attēlu kartēšanas risinājumi kā PIX4Dfields un Agremo, kuri piedāvā plašu funkcionalitātes klāstu daudzās lietojamību vidēs – bezsaišu, tīmekļa, Windows, macOS un ĢIS programmatūras kā QGIS, kas nodrošina pilno ĢIS pieredzi. Lai akurāti salīdzinātu risinājumus, darba autors izvirza vairākas pamata kritērijus ar kuriem salīdzināt un novērtēt līdzīgos risinājumus: sašūta ortofoto izveide no bezpilota gaisa kuģu attēliem, ortofoto vizualizācija kartē, sensoru datu integrācija un vizualizāciju kartē, sensoru datu pārskats attiecīgi izvēlētajam laikam un datumam, ortofoto kvalitātes konfigurācijas iespēja un sensora datu diagrammas pārskats.

## **1.1. PIX4Dfields risinājums**

PIX4Dfields ir bezpilota gaisa kuģa un satelīta attēlu kartēšanas Windows un macOS lietojumprogramma, kas nodrošina ražas analīzi un precīzu lauksaimniecību. Programmatūra atbalsta vairākas valodas kā: angļu, vācu, japāņu, spāņu, kīniešu, portugāļu, franču utt. Programmatūra ir maksas, sākot no 290 EUR mēnesī, bet darba autoram bija iespējams izmēģināt risinājuma darbību caur bezmaksas izmēģinājuma kontu, kas dod pamata funkcionalitāti 15 dienu periodā [3].

Sākot darbu ar PIX4Dfields, lietotājs spēj izvēlēties satelīta datus noteiktās robežās vai aerofotogrāfijas bezpilota gaisa kuģa uzņemtas bildes kartes GeoTIFF ģenerēšanas nolūkiem. Ir iespēja izvēlēties un apstrādāt jau izveidotu GeoTIFF failu. Pēc bezpilota gaisa kuģa bildes izvēles tiek ģenerēta GeoTIFF karte ĢIS vidē, kuru ir iespējams eksportēt līdzīgi ar citiem datiem kā slāniem, operācijām, robežām, anotācijām, statistikām, atskaitēm, momentuzņēmumiem un žurnālfailiem.

Ejot cauri lietojumprogrammai tiek piedāvāti 2 apstrādes opcijas: ātrā apstrāde – ātrāks rezultāts, laba ortomozaīka bet rupjš augstuma modelis, ideāls priekš izlūkošanas vai precīzā apstrāde – precīza ortomozaīka, detalizēts augstuma modelis un ideāls priekš mērījumiem, izmēģinājuma gabaliem un pacēluma analīzei. Aplikācija piedāvā slīdni attēla izšķirtspējas pielāgošanai, no kurās būs atkarīgs apstrādes laiks, tiek piedāvāti vairāki papildu iestatījumi kā GPU paātrinājums, radio metriskā korekcija u. c.

Ātrās apstrādes rezultējošā ortomozaīkas kartes kvalitāte neatšķiras drastiski no precīzās apstrādes, bet acīmredzama atšķirība ir redzama virsmas modeļa detalizācijā. Lietojumprogramma izvēlnē piedāvā indeksa kartes, zonēšanas un “magiskās” operācijas.

Salīdzinājumam būtiskākā funkcionalitāte ir veģetācijas indeksa kartes (VARI indeksa) izveidošana, kura ir pieejama caur indeksa izveides opciju. Radītais VARI indekss ir augstas kvalitātes un ļoti konfigurējams ar krāsas režīmiem, anotācijas izveides opcijām, caurredzamības iespēju, un dinamiskās diapazonu izmaiņas opciju, kas vizualizē atšķirību starp slāņa tumšākajām un spilgtākajām veģetācijas daļām.

PIX4Dfields ir viena no vispopulārākajām ražas un veģetācijas izmeklēšanas analīzes aplikācijām. Aplikācija dod dziļu ieskatu ražas izmeklēšanas operācijās, bet nespēj integrēt darba mērķa otro svarīgo komponenti – temperatūras un mitruma sensoru datus GIS vidē.

## 1.2. Agremo risinājums

Agremo ir lauka analīzes tīmekļa lietotne, kas piedāvā bezpilotu gaisa kuģa attēlu sašūšanu, analīzi, atskaites un izsmidzināšanas kartes. Serviss ir bezmaksas līdz lietotājs pieprasītu noteiktu apjomu analīzes. Pēc tam lietotājam būs nepieciešams iegādāties servisa abonementu, kura sākuma cena nav zināma, jo katrs abonements tiek pielāgots lietotāja vēlmēm un prasībām. [4]

Sākot tīmekļa lietotni lietotājs spēj izvēlēties demonstrācijas lauku, importēt jau izveidotu lauku, savienojoties ar trešo pušu MyJohnDeere lietojumprogrammu vai izveidot paša pielāgotu lauku.

Veidojot pašu pielāgotu lauku ir nepieciešams uzstādīt lauka nosaukumu, aprakstu, koordinātas, auga tipu un lauka īpašnieku. Tālāk lietotājam ir opcijas pievienot jaunu karti: pieprasīt bezpilotu gaisa kuģu lidojumu vai izmantot jau eksistējošus datus. Jaunā kartes izveidē ir nepieciešams uzstādīt kartes nosaukumu, datumu, kameras tipu un augšupielādēt ZIP failu ar neapstrādātiem bezpilotu gaisa kuģa attēliem vai ar jau eksistējošu ortofoto GeoTIFF failu, vēlams *EPSG:4326* projekcijā.

Uz izveidotās kartes lietotājam ir nepieciešams iezīmēt lauka robežu, lai turpinātu darbu ar karti. GIS skatā, izvēloties kādu no izveidotajiem laukuma poligoniem, lietotājs var:

- izvēlēties punktu kartē un iegūt tās koordinātas;
- izmērīt distanci starp vairākiem punktiem;
- iezīmēt poligona apgabalu un iegūt laukuma mēriju;
- veidot stenda uzskaites atskaiti norādot – analīzes nosaukumu, kultūras veidu, augšanas posmu, sēšanas blīvums, augi uz hektāru mēriju un komentāru;
- veidot auga veselības un nezāļu meklēšanas analīzi norādot – analīzes nosaukumu, kultūras veidu, augšanas posmu, komentāru/īss stresa aprakstu.

Auga veselības analīze parāda procentuālu sadalītu veselību diagrammā: lieliska veselība, laba veselība, slikta veselība un nav veģetācijas. Kā arī nezāles analīzē lietotājs redz nezāles izplatību kartē un nezāles līmeņu sadali sektorū līmeņos: tīrs apgabals, zems nezāļu līmenis un augsts nezāļu līmenis. Rezultāti tiek vizualizēti GIS vidē ar attiecīgām krāsām, kas norāda auga veselību un nezāļu līmeni.

Skaidri var redzēt, ka Agremo piedāvā noderīgas un spēcīgas analīzes, kas palīdz iegūt dziļāku ieskatu lauka un augu veselībā un stāvoklī. Kā arī Agremo piedāvā daudz mērišanas rīkus kā koordinātas, distances un laukuma mērišanu, bet tīmekļa lietotne nespēj integrēt jau eksistējošus temperatūras un mitruma sensora datus GIS vidē.

### 1.3. QGIS risinājums

QGIS (Kvantu ģeogrāfiskās informācijas sistēma) ir bezmaksas atvērtā pirmkoda programmatūra, kas specializējas ģeogrāfisko informācijas sistēmu apstrādē un darbībā, nodrošinot daudz attīstītas funkcionalitātes kā daudzu datu slāņu pārvaldi, lai vizualizētu GeoTIFF failus, rastra un vektoru datus, pacēlumu slāņus, mākoņu pārkājumu, satelīta skatu, kā arī GPX, XYZ, WCS slāņus. Šo programmatūru var vislabāk izskaidrot ar citātu no oficiālās dokumentācijas – “Tāpat kā mēs izmantojam tekstapstrādes programmu, lai rakstītu dokumentus un apstrādātu vārdus datorā, mēs varam izmantot GIS lietojumprogrammu, lai apstrādātu telpisko informāciju datorā. GIS apzīmē “Ģeogrāfiskās informācijas sistēma”.” [5]

Programmas galvenajā skatā ir redzams kartes skats un rīka josla ar daudziem iestatījumiem. Tā kā programmatūra ir GIS, tā var nodrošināt nepieciešamo GeoTIFF attēlošanas funkcionalitāti caur rastra slāņu izveidošanu. Līdzīgi, lai pievienotu ielas skata vai satelīta slāni, lietotājs spēj izveidot XYZ slāni. XYZ slāņa izveidē ir iespējams iestatīt kartes skata pakalpojuma sniedzējus: “OpenStreetMap”, “Mapzen Global Terrain” vai arī paša lietotāja uzstādītu risinājumu. Lietotājs spēj iestatīt XYZ kartes slāņa maksimuma un minimauma tālummaiņu, flīzes izšķirtspēju un interpolāciju, kā arī uzstādīt autentifikāciju priekš kartēšanas API.

Lai realizētu sensora datu integrāciju ir nepieciešams pārveidot esošos sensora JSON datus uz GeoJSON formātu, kas ir JSON objekts, kas satur datu tipu, īpašības, ģeometrijas tipu un koordinātas. Nepieciešamas arī ir izveidot jaunu datu struktūru specifiski ierīču koordinātas glabāšanai, jo oriģinālajos datos koordinātas nav iekļautas. Izveidoto GeoJSON failu lietotājs spēj vilkt un nomest QGIS sistēmā, kas pievienos datus esošajam kartes skatam, kur ir iespējams redzēt sensora datu vizualizāciju. Stilizēt datus tālāk ir iespējams ar graduētiem simboliem vai krāsu rampām, lai vizualizētu temperatūru, mitrumu utt.

QGIS ir strikti GIS programmatūra, kas piedāvā ļoti plašu GIS funkcionalitātes klāstu, kurš nav aktuālas uzdevumu ietvarā, kā arī QGIS nav spējīgs izveidot nepieciešamo gala lauka ortofoto no atsevišķiem bezpilota gaisa kuģu attēliem, bet tikai piedāvā jau esošu GeoTIFF faila apstrādi GIS vidē. Tāpēc darba autors uzsver, ka risinājums nav piemērots darba mērķa realizācijai.

Darba autors tabulā 2.1. apkopo risinājumu funkcionalitātes salīdzinājumu.

2.1. tabula

### Līdzīgu risinājumu salīdzinājums

| Funkcija   | PIX4Dfields   | Agremo   | QGIS  |
|--|---|--|---|
| <b>Ortofoto izveide no bezpilota gaisa kuģa attēliem</b> | Jā, ļoti vienkārša un optimizēta darbplūsma               | Nav paredzēts ortofoto izveidei, izmanto jau gatavus datus | Nav paredzēts ortofoto izveidei, izmanto jau gatavus datus  |
| <b>Ortofoto kvalitātes konfigurācija</b>                 | Vienkārša kvalitātes kontrole (izšķirtspēja, līdzināšana) | Nav paredzēts ortofoto izveidei, izmanto jau gatavus datus | Nav paredzēts ortofoto izveidei, izmanto jau gatavus datus  |
| <b>Ortofoto vizualizācija kartē</b>                      | Jā, draudzīga lietotāja vide                              | Vizualizē jau veidotu GeoTIFF ortofoto GIS vidē            | Spēcīga kartes vizualizācija ar esošu GeoTIFF failu vai fīzes serveri, bet sarežģītāka uzstādīšana    |
| <b>Sensoru datu integrācija un vizualizācija kartē</b>   | Neatbalsta sensora datu integrāciju                       | Neatbalsta sensora datu integrāciju                        | Brīva datu pievienošana un vizualizācija, bet jāformātē datus specifiskā formātā                      |
| <b>Sensora datu diagrammas pārskats</b>                  | Neatbalsta sensora datu integrāciju                       | Neatbalsta sensora datu integrāciju                        | Atbalsta datu diagrammas un grafikus (izmantojot grafiku spraudņus vai papildus rīkus)                |
| <b>Sensoru datu pārskats pēc laika un datuma</b>         | Neatbalsta sensora datu integrāciju                       | Neatbalsta sensora datu integrāciju                        | Ar pareizu konfigurāciju iespējama automātiska filtrēšana pēc laika (izmantojot laika slāņus/filtrus) |

Eksistē daudzi risinājumi, kas spēj nodrošināt darba uzdevuma pamata funkcionalitāti kā drona attēlu ortofoto izveidi un vizualizēšanu vai sensora datu integrāciju GIS kontekstā, bet tikai ar apjomīgu konfigurāciju. Taču neeksistē programmatūra, kas spēj nodrošināt visu nepieciešamo funkcionalitāti vienā apvienotā, viegli lietojamā risinājumā.

## **2. SISTĒMAS PRASĪBAS**

Darba mērķa realizācijai tiks veidota tīmekļa lietotne, kas nodrošina ērtu lietojamību, drošu sesijas darbību un intuitīvu darba plūsmu paļaujoties uz mākoņu tehnoloģijām darba progresu, sensoru datu un lietotāju izveidoto ortofoto glabāšanai. Aplikācija sastāvēs no vairākiem slāņiem un komponentēm. Tīmekļa vietni var sadalīt divos plašos iedalījumos: priekšpusē un aizmugursistēmā.

Tika izvēlēts tīmekļa vietnes sistēmas tips, jo tas nodrošina plašu pieejamību, datu uzglabāšanu un augstu lietojamību bez nepieciešamas instalācijas un konfigurācijas soļiem, kas ir ideāli pielietojumā lauksaimniekiem, kuriem iespējams nav plašas tehnoloģiskās zināšanas aplikācijas uzstādīšanā.

### **2.1. Eksistējošo ievada datu apraksts**

Ievada dati – sensoru dati un bezpilota gaisa kuģu attēli – tiek iegūti no lauka. Prototips tiek strukturēts atbilstoši ievada datu formātam, it īpaši sensoru datu API formātam.

#### **2.1.1 Bezpilota gaisu kuģa attēli**

Prototipa izstrādē izmantotie lauka attēli tika iegūti no bezpilota gaisa kuģi – DJI Phantom 4 Pro ar multispektrālo kameru. Uz lauka tika izvietoti četri kontrolpunkti – viens katrā stūrī, no kuriem pamata lidojuma programmatūra aprēķināja optimālās maršruta robežas un lidojuma trajektoriju, nodrošinot aptuveni 85% attēlu pārklājumu. Lidojumi tika veikti dienas vidū, saulainos apstākļos ar zemu mākoņu segumu, 10 metru augstumā, kas nodrošināja pietiekami augstu attēlu izšķirtspēju un samazināja kopējo lidojuma laiku. Rezultātā iegūtie attēli tika saglabāti JPG formātā ar iekļautām ģeogrāfiskā platuma un garuma koordinātām.

#### **2.1.2 Sensora dati**

Darba autors tika apgādāts ar temperatūras un mitruma sensoru datiem, kuri tiek glabāti Oracle serverī. Pašreizējās sistēmas prototipa lietotājs nespēj izvēlēties citu sensoru datu servera API. Dati tika ierakstīti no Vidzemes augstskolas studenta veidotām sensoru ierīcēm. Sensori tika sadalīti un novietoti atšķirīgos pacēlumos, lai pārklātu pēc iespējas vairāk teritoriju, kā arī sensori tika novietoti dažādās lauka daļās – daļā, kurā netiek aplaistīts lauks un daļā, kur tiek aplaistīts.

Serveris apkalpo sensora datus caur API, kuru pieprasījumu galapunkts seko sekojošam formātam: sensors/date/{datuma diapazonas sākuma datums}/{datuma diapazonas beigu datums}, kur diapazonas datums ir mēneša pirmie trīs burti angļiski,

mēneša diena un gads – piemēram, JAN202023. Iegūtie pieprasījuma dati JSON datu formātam, kas satur:

- sarakstu *items* – satur sensora ieraksta vienības (saraksta garums nepārsniedz 1000 ierakstus);
- būla vērtību *hasMore* – norāda, vai datumu diapazonā ir pieejami vēl vairāk sensoru datu ieraksti;
- vesela skaitļa vērtību *limit* – norāda pieprasījumu sensora ieraksta ierobežojuma skaitu;
- vesela skaitļa vērtību *offset* – norāda sensoru ieraksta skaitu nobīdi no oriģinālā pieprasījuma;
- vesela skaitļa vērtību *count* – norāda *items* saraksta vienības skaitu;
- sarakstu *links* – satur saites uz API pieprasījumiem, kas ir saistīti ar pašreizējo pieprasījumu, piemēram, *next* saite, kas norāda uz nākamo pieprasījuma *offset* soli gadījumā, ja pašreizējais pieprasījums nav atgriezis visus iespējamos sensora datus. *Next* saite ir vienīgais saraksta objekts, kas ir aktuāls darba izveidē, un saite uzstāda nākamo pieprasījuma nobīdi, izmantojot *?offset* URL vaicājumu (query), katra reizi palielinoties par 1000.

Sensora ieraksta vienības informācija satur sensora ID (*device id*), ieraksta datumu un laiku (*s\_date*), gaisa temperatūru un mitrumu (*air temperature*, *air humidity*), augsnes temperatūru 1, 2 un mitrumu 1, 2 (*soil temperature 1*, *soil temperature 2*, *soil moisture 1*, *soil moisture 2*). Gadījumā, ja ierakstā nav pieejams temperatūras vai mitruma mēriņums, attiecīgā vērtība ir uzstādīta uz *null*.

## 2.2. Funkcionālās prasības

1. Jānodrošina lietotāja autentifikācija:
  - 1.1. lietotājs spēj pierakstīties tīmekļa vietnē ar Google kontu (caur Google autentifikācijas platformu);
  - 1.2. sistēma izvada paziņojumu un noraida pierakstīšanos, ja lietotāja e-pasta adrese nav pilnvarota sistēmas darbībai;
  - 1.3. ja lietotājs nav datubāzes lietotāju sarakstā un e-pasts ir pilnvarots, tiek izveidots jauns lietotāja korts;
  - 1.4. lietotājs spēj manuāli izrakstīties no tīmekļa lietotnes.
2. Jānodrošina lietotāja konta izveide:
  - 2.1. tiek pārbaudīts, vai lietotāja e-pasta adrese ir pilnvarota sistēmas darbībai caur datubāzes tabulas vaicājuma;
  - 2.2. jaunam lietotājam tiek izveidots atsevišķs WebODM projekts;

- 2.3. izveidotais WebODM projekta ID tiek saistīts ar lietotāja kontu datubāzē un saglabāts sīkdatnē;
  - 2.4. lietotāja korts (e-pasta adrese) tiek saglabāts datubāzē .
3. Jānodrošina tīmekļa lietotnes navigācijas joslu funkcionalitāti:
    - 3.1. joslā lietotājs var izmantot izrakstīšanās pogu;
    - 3.2. joslu var paslēpt caur pogas spiedienu;
    - 3.3. josla satur visas pieejamās tīmekļa vietnes lapas.
  4. Jānodrošina ortofoto izveide no bezpilota gaisa kuģu attēliem:
    - 4.1. katrs autentificēts lietotājs spēj izveidot jaunu ortofoto;
    - 4.2. kartes nosaukumam ir jābūt no 1 līdz 100 rakstzīmes garam;
    - 4.3. ortofoto izšķirtspēja tiek regulēta ar slīdnī no 0.0 līdz 5.0;
    - 4.4. lietotājs spēj izvēlēties ZIP failu, kas satur vairākus bezpilota gaisa kuģa JPG attēlus augšupielādei;
    - 4.5. ZIP faila augšupielādē tiek parādīta augšupielādes progresā josla;
    - 4.6. ortofoto izveidē tiek parādīta izveides progresā josla;
    - 4.7. lietotājs spēj atcelt ortofoto augšupielādi vai izveidi.
  5. Jānodrošina lietotāja veidoto ortofoto kontrole:
    - 5.1. lietotājs spēj pieklūt tikai pie paša veidotām ortofoto kartēm;
    - 5.2. visi lietotāja veidotie ortofoto ir redzami sarakstā no WebODM pieprasījuma;
    - 5.3. sarakstu ir iespējams atjaunināt.
  6. Jānodrošina lietotāja veidoto ortofoto datu apstrādi:
    - 6.1. ortofoto var izdzēst;
    - 6.2. ortofoto var lejupielādēt GeoTIFF faila formātā;
    - 6.3. ortofoto var atvērt kartes skatā;
    - 6.4. lietotājs var atvērt lokālu GeoTIFF failu un redzēt to kartes skatā.
  7. Jānodrošina ortofoto kartes skatu:
    - 7.1. spēja izvēlēties sensora datu datumu;
    - 7.2. spēja izvēlēties sensora datu laiku;
    - 7.3. spēja redzēt izvēlēto koordinātu kartē;
    - 7.4. spēja iziet no kartes skatu atpakaļ uz ortofoto sarakstu;
    - 7.5. karte tiek attēlota ar visiem ortofoto un sensoru datiem.
  8. Jānodrošina sensora koordināšu datu apstrādi:
    - 8.1. sistēma paziņo visus sensora ID, kuriem ir nepieciešams iestatīt koordinātas un iespējamas mainīt vai izdzēst koordinātas;

- 8.2. spēja uzstādīt koordinātu sensoram;
  - 8.3. spēja izmainīt jau uzstādītu sensora koordinātu;
  - 8.4. spēja izdzēst sensora koordinātu;
  - 8.5. spēja izvēlēties iestādāmo sensoru no nolaižamas izvēlnes;
  - 8.6. spēja atcelt koordinātu izvēles procesu;
  - 8.7. koordināta tiek izvēlēta spiežot uz kartes.
9. Jānodrošina ortofoto un sensora datu integrāciju kartes skatā:
    - 9.1. spēja vizualizēt ortofoto kartē;
    - 9.2. spēja vizualizēt VARI kartē;
    - 9.3. sensora dati tiek vizualizēti sensoriem, kuriem ir iestatītas koordinātas;
    - 9.4. spēja redzēt sensoru datus vairākos informācijas slāņos: gaisa temperatūra, gaisa mitrums, augsnes temperatūra 1, 2 un augsnes mitrums 1, 2;
    - 9.5. spiežot uz vai novietojot cursoru virs sensoru pārklājuma informācijas slāni, ir iespējams redzēt mēriņuma vērtību un mērvienību;
    - 9.6. spiežot uz sensora ierīces apli, tiek attēlota sensora ID un koordinātu vērtība.
  10. Jānodrošina sensoru datu pārlūkošanu:
    - 10.1. spēja izvēlēties sensora datu datumu diapazonu;
    - 10.2. spēja pārlūkot cauri sensora datiem noteiktā diapazonā;
    - 10.3. spēja izvēlēties mēriņumu veidu caur nolaižamās izvēlnes;
    - 10.4. datu vizualizācija diagramma;
    - 10.5. spēja pārskatīt sensoru datu diagrammu izvēlētajā ortofoto datumā;
    - 10.6. spēja aplūkot individuāla mēriņuma informāciju.

### **2.3. Nefunkcionālās prasības**

1. Sistēmas saskarnes valodai ir jābūt latviešu valodā.
2. Sistēma ir pieejama tumšā vai gaišā tēmā atkarībā no lietotāja datora sistēmas iestājumiem.
3. Risinājums ir tīmekļa vietne, kurai var piekļūt caur interneta pārlūku.
4. Tīmekļa vietne ir veidota Python programmēšanas valodā.
5. Tīmekļa vietnei ir divas daļas: aizmugursistēma un priekšpuses sistēma:
  - 5.1. tiek integrēts aizmugursistēmas WebODM API caur pieprasījumiem;
  - 5.2. API pieprasījumiem ir jābūt pilnvarotiem caur žetonu autentifikāciju.
6. Datubāzes pieprasījumiem tiek izmantoti sagatavoti vaicājumi, lai novērstu SQL injekciju.

7. Tīmekļa vietne vienlaikus spēj atbalstīt no 1 līdz 5 lietotājus, kuri vienlaikus nepieprasā ortofoto izveidi.
8. Lietotāja WebODM API žetons un projekta ID tiek šifrēti, pielietojot *EncryptedCookieManager*.
9. Tīmekļa vietnes sensitīvie dati kā datubāzes un WebODM paroles tiek glabātas drošā vidē – *secrets.toml* failā, kas nav pieejams lietotājiem.
10. Tīmekļa vietnes darbībai ir nepieciešama lietotāja autentifikācija caur Google autentifikācijas platformu.
11. Tīmekļa vietnes lapas, kā arī API pieprasījumi ielādējas ne ilgāk kā 2 sekundes.
12. Tīmekļa vietne koncentrējas uz darba plūsmu galda vai portatīvo datoru vidē, bet ir funkcionāli lietojams mobilajās ierīcēs.

### **3. TEHNOLOGISKIE RISINĀJUMI UN TO SALĪDZINĀJUMS**

#### **3.1. Priekšpuses tehnoloģijas**

Aplikācijas priekšpuse ir atbildīga par funkcionalitāti ar ko lietotājs tieši mijiedarbosies kā tīmekļa lietotnes lapām un visām komponentēm, kas tajā tiek iekļautas – navigācijas josla, informācijas teksti, visu veidu pogas un interaktīvās komponentes, GIS karte, kura vizualizēs integrētos sensora un bezpilota gaisa kuļa datus un visbeidzot sensora datu diagrammas. Priekšpusē tiks izmantots apvienojošs ietvars kurš būs atbildīgs par visām iepriekš minētajām funkcionalitātēm. Prototipa priekšpuses izveidē tiek izmantota Python programmēšanas valoda pēc zinātniskās vadītājas pieprasījuma.

##### **3.1.1 Lietotāju saskarnes un datu apstrādes ietvars**

Priekšpusē tiks izmantots atvērtā pirmkoda Python programmēšanas valodas lietotņu ietvars – Streamlit dēļ tās iebūvētās interaktivitātes un datu apstrādes spējas. Esošo Python tīmekļa vietnes izstrādes ietvaru salīdzinājumā [6] var redzēt, ka Streamlit ir izstrādāts, domājot par vienkāršību, ļaujot izstrādātājiem tikai dažu minūšu laikā pārvērst datu skriptus koplietojamās tīmekļa lietotnēs. Streamlit izmanto *st.dataframe* efektīvai datu apstrādei, kas ir optimizēta lieliem datu kopumiem, kas ir aktuāla priekšrocība sensoru datu apstrādē *st.dataframe* funkcijā. Streamlit izceļas ar interaktīvu datu lietojumprogrammu izveidi, izmantojot minimālu standarta kodu. Šī interaktivitāte ir būtiska koordinātas uzstādīšanā un lietotāja mijiedarbības efektīvai uztveršanai.

Streamlit ir vairākas priekšrocības pāri tradicionāliem Python tīmekļa vietnes izstrādes ietvariem kā Flask vai Django:

- ātrāka iestatīšana un aplikācijas izstrādes process, bet grūtākas konfigurēšanas iespējas;
- iebūvēts stils un navigācijas sistēma;
- iebūvēts Google OAuth 2.0 autentifikācijas atbalsts;
- neskaitāmi daudz noderīgas interaktīvas komponentes;
- sesijas un ilgtermiņa mainīgo pārvalde;
- kešatmiņas metožu atbalsts, efektīvai datu apstrādei;
- diagrammas un datu kopu vizualizēšanas risinājumi.

Šīs ietvara pieejas lielākā vajība ir lietotāja interfeisa konfigurēšanas trūkums, bet darba mērķa kontekstā tas nav būtisks šķērslis funkcionalitātes realizācijā un ir iespējams

pārvarēt šo trūkumu ar *st.markdown* funkcijas pielietojumu, kas spēj injicēt CSS īpašības HTML kodā, tādejādi nodrošinot nepieciešamo stila pielāgojamību. Streamlit atbild par visaptverošām darbībām kā navigāciju, tīmekļa lietotnes struktūru, autentifikāciju un satura renderēšanu, bet specifiskām vajadzībām, kā GIS kartes un datu diagrammas vizualizēšanu, ir nepieciešams izmantot papildus Python pakotnes.

### 3.1.2 Kartēšanas pakotne

Streamlit ietvarā, lai vizualizētu GeoTIFF karti, kā arī sensora datus, ir nepieciešams izmantot kādu no kartēšanas Python pakotnēm. Dēļ pakotnes interaktivitātes atbalstu un lietošanas vienkāršību, darba autors izvēlējās Python pakotni – Folium [7]. Folium pakotnei pastāv Streamlit integrācijas risinājums – pakotne *streamlit\_folium*, kas spēj renderēt karti tieši Streamlit aplikācijā un piedāvā papildus funkcionalitāti kā kartes vērtības atgriešanu, kas ļauj tieši Python kodā piekļūt pie izvēlētām koordinātām, robežām, pietuvināšanas līmeni, pēdējo noklikšķināto objektu, kartes centru utt. Folium arī piedāvā plašu funkcionalitātes klāstu, kas ir nepieciešams tīmekļa vietnes izstrādē:

- pietuvināšanas konfigurācija;
- specifiska kartes flīžu konfigurācija priekš satelīta flīžu skata;
- atsevišķu datu slāņu izveide un kontrole;
- datu vizualizēšana apļa formātā ar konfigurējamu krāsu, necaurredzamības kontroli un uznirstošu logu datu īpašības parādīšanai;
- GeoTIFF faila atbalsts;
- statiska un interaktīva karte, kas spēj atgriezt koordinātas no lietotāja mijiedarbības.

Eksistē citi lieliski risinājumi, kas ir spējīgi veikt lielāko daļu no prasībām kā Plotly, Ipyleaflet bet šie risinājumi nav interaktīvi un nespēj nodrošināt vienu no tīmekļa lietotnes pamata darbībām – koordinātas iegūšanu no lietotāju mijiedarbības. Geemap ir vēl viens lielisks GIS risinājums, kas izmanto GEE – Google Earth Engine, bet Streamlit ietvara kontekstā, tas ir sarežģītāk uzstādīt un ir prasa daudz pakotņu atkarības, kas palielina servera slodzi.

### 3.1.3 Sīkdatņu atbalsts

Ilgtermiņa sīkdatnes glabāšanai darba autors izmanto *st\_cookies\_manager* pakotnes *EncryptedCookieManager* funkcionalitāti, kas nodrošina sīkdatnes integrāciju Streamlit Python kodā. Noklusējumā Streamlit ietvars nespēj komunicēt ar pārlūka sīkdatni, kas ir nepieciešams ilgtermiņu datu uzglabāšanā – lietotāja WebODM projekta ID un WebODM autorizācijas piekļuves žetonu. *EncryptedCookieManager* sīkdatnes pārvaldei ir svarīga

priekšrocība – kad tiek iestatīta sīkdatne, saturs tiek šifrēts. Tas ir it īpaši svarīgi glabāšanai konfidenciāliem datiem kā pieklubes žetoniem, jo, kad tīmekļa lietotne tiks izvietota, tā būs pakļauta ārējiem draudiem, šifrēšana nodrošina, ka pat, ja dati tiek apdraudēti, uzbrucējam tie nav derīgi.

### 3.1.4 Datubāzes savienojums

Pakotne *psycopg2* tiek izmantota, datubāzes savienojuma izveidei ar PostgreSQL datubāzi. Kā arī pakotnes *sql* funkcija nodrošina SQL vaicājumu sagatavošanu, kas pasargā tīmekļa vietni no SQL injekcijas uzbrukumiem.

### 3.1.5 GeoTIFF faila apstrāde

GeoTIFF failu datu nolasīšanai, koordināšu sistēmas izmaiņai, koordinātas centra un robežas aprēķināšanai tiek izmantotas *Rasterio* un *pyproj* pakotnes, kā arī *NumPy* un *PIL/Pillow* pakotnes. GeoTIFF attēla pikselu operācijām attēlu tukšo datu noņemšanu vajadzībām un gala PNG faila izveidei. PNG faila izveide ir nepieciešama Folium kartes darbībai, jo GeoTIFF formāts tieši netiek atbalstīts.

### 3.1.6 Citas pakotnes

Darba autors izmants *branca.colormap* pakotni, atšķirīgām temperatūras un mitruma pārklājumu krāsu aprēķināšanai. Vispārējai pieprasījumu sūtīšanai un specifiski veidotā WebODM pieprasījumu risinājumā, kas atkārtoti sūta pieprasījumu nelabā pieklubes žetona gadījumā tiek izmantota *requests* pakotne. Sensoru datu apstrādei un attēlošanai tiek izmantota *pandas* pakotne, kas izveido datu rāmi priekš *altair* pakotnes diagrammas izveides.

## 3.2. Aizmugursistēmas tehnoloģijas

No otras pusēs, aizmugursistēma ir atbildīga par visu funkcionalitāti, ko lietotājs tieši nerēdz:

- autentifikācijas komponentes;
- datubāze;
- temperatūras un mitruma sensoru datu serveris;
- bezpilota gaisa kuļa attēlu ortofoto apstrādes serveris;

### 3.2.1 Datubāze

Tā kā tīmekļa vietne neveic sarežģītas operācijas, skatus, trigerus vai procedūras, galvenais un vissvarīgākais veikspējas mērījums ir atlasīšanas (SELECT) darbības ātrums. S.V. Salunke un A. Ouda publikācijā [8] tika izmērīta PostgreSQL un MySQL datubāžu veikspēja, izmantojot atlases un ievietošanas vaicājumus zem primāriem un sarežģītiem apstākļiem, simulējot reālās pasaules scenārijus. Mērījum rezultātā pētījums [8] konstatēja,

ka PostgreSQL datubāze ir izcila atlasīšanas darbībās – PostgreSQL izpildes laiks 1 miljonam ierakstu bija no 0,6 ms līdz 0,8 ms, savukārt testi fiksēja MySQL ātrumu no 9 ms līdz 12 ms, kas norāda, ka PostgreSQL ir aptuveni 13 reizes ātrāks. Līdzīgos *WHERE* vaicājuma mērījumos tika novērots, ka PostgreSQL bija 9 reizes ātrāks nekā MySQL. Autori pēc mērījumu veikšanas un analīzes secina un uzsver PostgreSQL datubāzes pārākumu: “These findings underscore PostgreSQL’s suitability for environments requiring low data latency and robust concurrent processing capabilities...” [8, 1 lpp.].

Balstoties uz apskatīto pētījumu, datubāzes nolūkiem tiks izmantota PostgreSQL datubāze – jaudīga, atvērtā koda objektu relāciju datubāzes sistēma ar vairāk nekā 35 gadu aktīvu attīstību, kas tai ir nopelnījusi spēcīgu reputāciju ar uzticamību, funkciju noturību un veikspēju [2]. Datubāze sastāv no vairākām tabulām: apstiprinātās e-pastu tabula, kura glabā apstiprinātās e-pasta adreses, kuras ir pilnvarotas izmantot tīmekļa lietotni un lietotāju tabula, kurā glabājas pilnvaroti lietotāji, kuriem tika izveidots WebODM projekta ID, lietotājiem tiek glabāts e-pasts un WebODM projekta ID.

### 3.2.2 Ortofoto izveides API

Tā kā kartes izveidošanas risinājums no individuāliem bezpilota gaisa kuģu attēliem ir ārpus šī darba ietvara, ir nepieciešams izvēlēties attiecīgu rīku, kas spēj atgriezt sašūtas ortofoto datus. Šim nolūkam galvenie pretendenti bija atvērtā koda programmatūras risinājumi NodeODM un WebODM, kas ir API implementācija no ODM ar papildus funkcionalitātēm. Abiem risinājumiem ir līdzīgs funkcionalitātes klāsts, bet ar svarīgām atšķirībām, kuras noteiks beigu tehnoloģijas izvēli.

NodeODM ir standarta API specifikācija gaisa attēlu apstrādei ar ODM dzinēju. API izmanto tādi klienti kā WebODM, CloudODM un PyODM [9]. Lietotājs spēj izveidot un ģenerēt ortofoto bezpilota gaisa kuģu attēliem caur API pieprasījumiem, un lejupielādēt visus gala failus vienā ZIP failā. Risinājums koncentrējas uz vienkāršību, nodrošinot tikai pamata funkcionalitāti ortofoto apstrādes procesā un darba gaitā. Lietotājs spēj konfigurēt kartes izveides procesu caur iestatījumiem, kuri pielāgo gala faila, kvalitāti, izšķirtspēju, punktu mākoņu kvalitāti, 3D modeļa detalizācijas līmeni, minimālo punktu skaitu katrā attēlā utt.

Nemot vērā, ka NodeODM ir viegla veida API lietotne, tā nenodrošina ilgstošu datu uzglabāšanu, autentifikācijas sistēmu, kartes grupēšanu un pārvaldes sistēmu, augu veselības indeksa un citas kartes izveidi. Kā arī nav iespējams lejupielādēt specifiskus resursus kā ortofoto, tikai visus resursus kopā vienā ZIP failā, kas padara datu apstrādi grūtāku Python vidē.

WebODM ir lietotājam draudzīga, komerciāla līmeņa programmatūra bezpilota gaisa kuģu attēlu apstrādei. Programmatūra nodrošina ģeoreferences kartes, punktu mākoņus, augstuma modeļus un teksturētu 3D modeļus ģenerēšanu. Tas atbalsta vairākus apstrādes dzinējus, pašlaik ODM un MicMac [10]. WebODM nodrošina visu funkcionalitāti, kas tika pieminēta NodeODM aprakstā, jo WebODM operācijas apstrādei izmanto NodeODM.

Šis risinājums ir pilni funkcionāla tīmekļa lietotne un API, kas piedāvā autorizācijas sistēmu, kartes grupējumu izveidi un projekta pārvaldes sistēmu, kartes izveides procesa novērošanu, ilgtermiņa datu uzglabāšanu utt. Galvenās iespējas, kuras ir svarīgas darba mērķa realizācijai ir ilgtermiņa datu glabāšana un projekta pārvaldes sistēma. Ilgtermiņu datu glabāšana ir svarīga, jo tās esamība API risinājumā nodrošina gan ar ortofoto izveidi gan failu servera funkcionalitāti, kas spēj glabāt un nodrošina ilgstošu piekļuvi ģenerētajiem failiem. Individuālas kartes grupēšana atsevišķos projektos ir svarīga funkcionalitāte, jo tā ļauj darba autoram izveidot jaunu, unikālu WebODM projektu katram lietotājam, kas sadala un organizē kartes datus pēc lietotāja. Šī pieeja ļauj lietotājam piekļūt tikai pie savām veidotajām kartēm.

Dēļ WebODM ilgstošās datu glabāšanas spējas, projekta pārvaldes sistēmas un auga veselības kartes ģenerēšanas funkcionalitātes, darba autors izvēlējās izmantot WebODM risinājuma realizācijā.

### 3.2.3 OIDC pakalpojumu sniedzējs

Streamlit ietvars piedāvā lietotāju autentifikāciju tīmekļa vietnē caur OpenID savienojumu (OIDC) protokolu, kas ir autentifikācijas protokols veidots uz OAuth 2.0 bāzes. OIDC nodrošina lietotāja autentifikāciju, bet ne autorizāciju. Autorizāciju – lietotāja piekļuves līmeņa pārvaldi (vai lietotājs tiek sistēmā vai nē), sistēma apstrādā pēc pierakstīšanos caur OIDC servisu. Autentifikācijas OIDC pakalpojumu sniedzēji, ko Streamlit atbalsta un iesaka ir: Google Identity, Microsoft Entra ID, Okta, Auth0.

Darba autors tīmekļa vietnes autentifikācijas nolūkiem izvēlējās Google OAuth 2.0 servisu. Tā kā Google e-pasta konti ir vispopulārākie pasaulei, sasniedzot vismaz 1,5 miljardu e-pasta adreses [11], tas nozīmē, ka lietotājiem, visticamāk, ir šāds korts. Apvienojot šo popularitāti ar Google OAuth 2.0 platformas pierakstīšanās procesa vienkāršību – lietotājam ir jāpierakstās tikai caur jau eksistējošu Google kontu bez vajadzības atcerēties papildus paroli, padara Google OAuth 2.0 par ideālu autentifikācijas risinājumu. Kā arī pētījums [12] norāda, ka OAuth 2.0 arhitektūra atbalsta augstāku mērogjamību un zemāku latentumu, kas ir piemērots vidēm, kurās prioritāte ir efektivitāte, kā arī šī tehnoloģija balstās uz

autentifikācijas slāni, kas pārbauda lietotāju identitāti, tādējādi uzlabojot drošību lietojumprogrammās, kurās nepieciešama lietotāja verifikācija.

## **4. SISTĒMAS IZSTRĀDE UN TESTĒŠANA**

Šajā nodaļā darba autors dos ieskatu izstrādes procesā un aprakstīs un tīmekļa vietnes galvenās sistēmas daļas, tās komponentes, aprakstu, darbības struktūru, sastāvdaļas, bloku shēmas, savienojumus utt.

### **4.1. Datubāzes sistēma**

Datubāzes sistēma izmanto jauktu piegājienu, apvienojot tradicionālu datubāzes sistēmu – PostgreSQL kopā ar jau eksistējošu WebODM API objektu struktūru, lai nodrošinātu ilgtermiņa datu glabāšanu, vienlaikus minimizējot sistēmas sarežģītību.

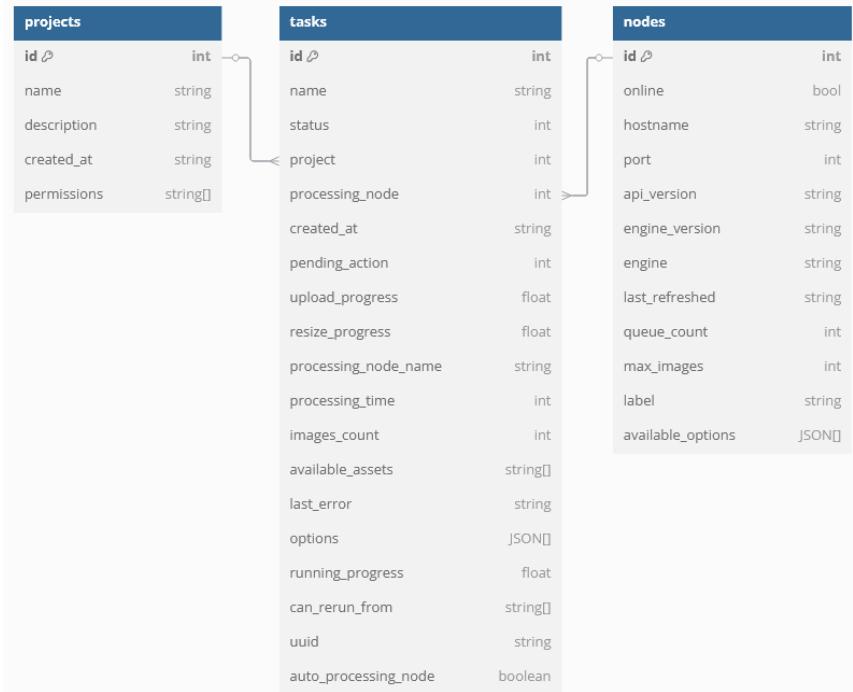
WebODM API nodrošina ortofoto izveidi no bezpilota gaisa kuģu attēliem un sastāv no 3 objektiem, kuri ir aktuāli darba autora sistēmas izstrādē. Struktūra (5.1. att.) tika izveidota, sekojot WebODM oficiālo dokumentāciju [13].

Lai izpildītu darba prasību, tīmekļa vietne, darbojoties caur WebODM API kā WebODM administratora lietotājs, izveido unikālu projektu (project), kas eksistē un ir saistīts ar katru unikālu lietotāju PostgreSQL datubāzē. WebODM projekts ir uzdevumu vienību kopums un sastāv no 5 atribūtiem, kuri apraksta projekta informāciju – ID, projekta izveides datumu (created\_at), nosaukumu (name) un aprakstu (description) un operācijas klāstu, ko pašreizējais lietotājs spēj veikt. Pierakstīšanās procesā lietotāja unikālais projekta ID tiek ielādēts no datubāzes Streamlit sesijas atmiņā, kas tālāk tiks izmantots WebODM API pieprasījumos.

Katrs lietotāja projekts sastāv no WebODM uzdevumiem (tasks), kuri ir bezpilota gaisa kuģu attēlu apstrādes vienības. Lietotāja veidota bezpilota gaisa kuģu attēlu ortofoto ir uzvedums, kas ir veidots ar ortofoto izveides opciju un ir saistīts ar lietotāja attiecīgo projektu. Uzdevumi sastāv no nosaukumu, ortofoto izstrādes stadijas statusu, pieejamiem resursiem, izvēlētajām datu apstrādes opcijām, apstrādes progresu un citiem mazāk svarīgiem atribūtiem, kurus darba autors tieši neizmanto tīmekļa vietnē.

Katram uzdevumam tiek automātiski piesaistīts apstrādes mezgls (node) – konteineris, kurš veic faktisko bezpilota gaisa kuģu attēlu apstrādi, lai iegūtu rezultējošo ortofoto, VARI, ortofoto flīzes un citus resursus. Apstrādes mezgli ir saistīti ar uzdevumiem un rūpējas par ievades attēlu apstrādi. Apstrādes mezgli ir datori vai virtuālās mašīnas, kurās darbojas NodeODM vai jebkura cita API, kas ar to ir saderīga. WebODM API ir uzstādīts ar noklusējuma Docker iestatījumiem, kurš izmanto vienu iebūvētu apstrādes mezglu – node-odm. Tīmekļa vietnes nolūkiem tiks izmantots šis vienīgais iebūvētais apstrādes mezgls, kas būs atbildīgs par visu uzdevumu attēlu apstrādi. Bet ir iespējams pievienot vairākus mezglus priekš vairāku uzdevumu paralēlu apstrādi dažādos mezglos, kas nodrošina aplikācijas

mērogojamību, taču tīmekļa vietnes darbības sfēra pieprasītikai vienu vienlaikus ortofoto izveidi, ko noklusējuma apstrādes mezgls var nodrošināt ar pietiekami ātru datu apstrādi.



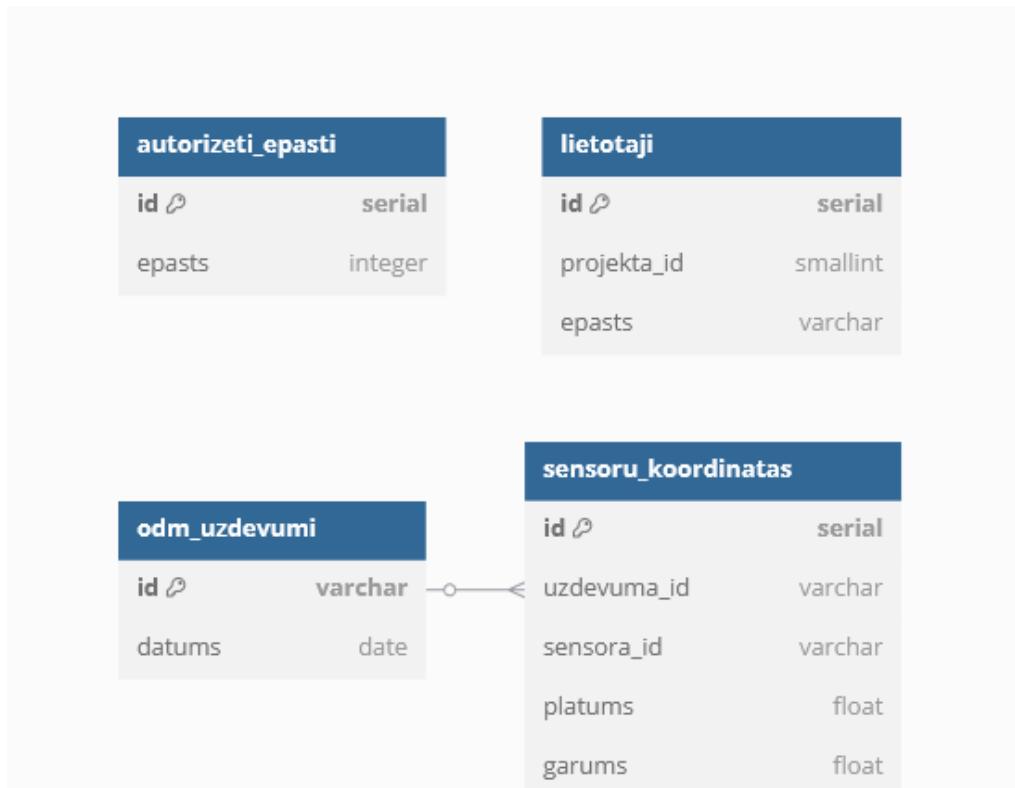
### 5.1. att. WebODM API objektu struktūra

PostgreSQL datubāze sastāv no 4 tabulām (5.2. att.):

1. Autorizēti e-pasti – sastāv no 2 kolonām, automātiski generētu ID un e-pastu. E-pasta kolona ir lietotāja Google konta e-pasta adrese, kura ir pilnvarota tīmekļa vietnes sistēmas darbībai. Lietotājs, kuram ir pilnvarota e-pasta adrese, spēj pierakstīties un izveidot tīmekļa vietnes kontu. Ja lietotāja e-pasts nav šajā datubāzes tabulā, sistēmas pierakstīšanās tiek noraidīta.
2. Lietotāji – tabulā tiek glabāti izveidoti lietotāja konti, ja Google konta e-pasta adrese ir pilnvarota. Tabula satur 3 kolonas: automātiski generētu ID, projekta ID, kas ir tieši saistīts ar WebODM projektu, kurš tiek izveidots katra jauna lietotāja tīmekļa vietnes pierakstīšanās gadījumā un lietotāja e-pasta adrese. Lietotāju tabulas ieraksti nav tieši saistīti ar autorizētu e-pastu tabulas ierakstiem ar saiti, lai administrators gadījumā spētu izdzēst autorizētu e-pasta tabulas ierakstu tajā pašā laikā paturot lietotāja ierakstu.
3. ODM uzdevumi – sastāv no 2 kolonām: ID, kurš ir tieši saistīts ar WebODM uzdevuma vienību un datumu. Datums nosaka WebODM uzdevuma bezpilotu gaisu kuģu attēlu uzņemšanas datumu, kas sistēma tiks izmantots, lai iegūtu attiecīgos sensora datus. Tabulas izveide bija nepieciešama, lai sistēma varētu ilgstoši izsekot no kura datuma pieprasīt sensora datus. ODM

uzdevuma ieraksts reprezentē lauku konkrētā datumā uz kura iespējams pastāv sensora ierīces, tāpēc tabulai ir viena pret daudziem saistība ar sensora koordinātām.

4. Sensoru koordinātas – tabula glabā labības lauka (ODM uzdevuma) sensoru koordinātas. Tabula sastāv no 5 kolonām: automātiski generētu ID, WebODM uzdevuma ID ar kuru sensora koordināta ir saistīta, sensora ID un paša koordināta, kas sastāv no ģeogrāfisko platumu un garumu. Tabula nodrošina ilgstošu sensora koordinātu glabāšanu katram uzdevumam (labības laukam) konkrētā datumā.



5.2. att. PostgreSQL datubāzes ER diagramma

## 4.2. WebODM API savienojums

WebODM API nodrošina bezpilota gaisa kuģu attēlu ortofoto un VARI kartes izveidi. Kā arī serviss nodrošina ilgstošu ortofoto kartes uzglabāšanu lietotāju projektos. Tīmekļa vietne nespēj darboties bez derīga WebODM savienojuma, tāpēc darba autors uzstādīja savienojumu sekojot oficiālo dokumentāciju [13]. Lai izmantotu API, ir nepieciešams autentificēties un izmantot iegūto JWT žetonu katra pieprasījuma galvenē. JWT žetons ir derīgs tikai 6 stundas pēc žetona izsniegšanas. Pēc 6 stundu perioda pieprasījumi, kuri satur žetonu ar beigušos derīguma termiņu, atgriež 503 statusa kodu ar ziņojumu “Signature has expired.”.

Ja WebODM autorizācijas galvene nav iestatīta sīkdatnē, Streamlit ietvars veic API pieprasījumu uz *token-auth* API galapunktu, izmantojot WebODM administratora autorizācijas datus (5.1. koda fragments). Veiksmīga pieprasījuma gadījumā JWT žetonu Streamlit ietvars saglabā sīkdatnē konkrētā formātā – {"Authorization": "JWT {žetons}"}. Gadījumā, ja JWT netiek veiksmīgi atgriezts, lietotājs nespēj turpināt tīmekļa vietnes operācijas un saskarne paziņo lietotājam par klūdu, piedāvājot iespēju atkārtoti pieprasīt WebODM autentifikācijas galveni.

```
def iestatit_galveni():

    try:
        atb = requests.post(
            f"{st.secrets.odm_url}/token-auth/",
            data={
                'username': st.secrets.odm_username,
                'password': st.secrets.odm_password
            }) # data atribūtā iestāda WebODM autorizācijas datus
        atb.raise_for_status() # Pārbauda pieprasījums ir veiksmīgs
        rez = atb.json()
        if rez["token"]:
            st.session_state.sikdatne["galvene"] = json.dumps({
                "Authorization": f"JWT {rez['token']}"})
        return True # Žetons tiek iestatīts sīkdatnē "galvene" vērtībā
    except Exception as e: # Ja pieprasījums nav veiksmīgs, klūda tiek attēlota
        st.error("Klūda WebODM savienojumā")
```

### 5.1. koda fragments. WebODM galvenes iestatīšanas funkcija

Sīkdatnes WebODM autorizācijas galvene, kā arī visi citi sīkdatnes dati ir pieejami *st.session\_state* mainīgajā – veids, kā koplietot mainīgos starp atkārtotām palaišanām katrai lietotāja sesijai, šis mainīgais tiek glabāts ilgstoši (līdz tīmekļa lapas pārlādes) un ir pieejams jebkurā koda daļā vai Streamlit failā, kur ir definēta Streamlit pakotne.

Darba autors izveidoja WebODM API pieprasījuma apstrādes metodi – *pieprasit\_odm* (pielikums Nr. 2), kas apstrādā visus WebODM pieprasījumus un pēc neveiksmīga 403 statusa koda pieprasījuma (pieķuve pieprasītajam resursam ir aizliegta, jo JWT žetona derīguma termiņš ir beidzies), un uzstāda WebODM autorizācijas JWT žetonu sīkdatnē un atkārtoti pieprasa neveiksmīgo API pieprasījumu, izmantojot jauni iegūto autentifikācijas žetonu. API pieprasīšanas metode spēj sūtīt GET un POST pieprasījumus atšķirībā no metodes uzstādītajiem parametriem. Katra pieprasījuma *headers* parametrs tiek automātiski ielādēts no *st.session\_state* vērtības.

### 4.3. Autorizācijas sistēma

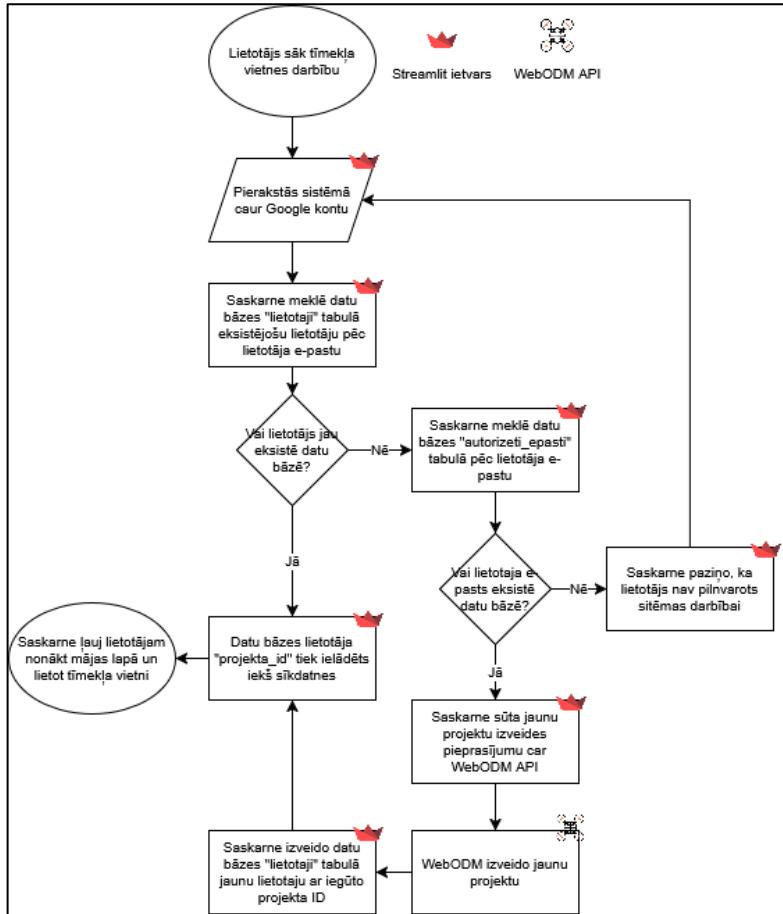
Lietotāja autorizācijas plūsma (5.3. att.) sākās ar neautentificēta lietotāja sistēmas darbības uzsākšanas. Ja Streamlit lietotāja informācijas objekts *st.user* nesatur patiesu *is\_logged\_in* vērtību, lietotājs automātiski tiek novirzīts uz autentifikācijas logu, kur, nospiežot pierakstīšanās pogu, sākās Google OAuth 2.0 autentifikācijas process. Koda fragments 5.2. demonstrē, izmantojot *st.login* funkciju, Streamlit ietvars automātiski sāk Google OAuth 2.0 procesu. Process nolasa aplikācijas *.streamlit* direktorijas *secrets.toml* failu, kur tiek uzstādīti Google autentifikācijas konfigurācijas mainīgie: *client\_id*, *client\_secret*, *cookie\_secret*, *redirect\_uri* un *server\_metadata\_url*. Streamlit ietvars gaida šādi nosauktus mainīgos, pēc kuru iestatīšanas Streamlit spēj komunicēt ar Google autentifikācijas servisu.

```
if not st.user.is_logged_in:  
    st.header("Tīmekļa vietne ir privāta")  
    st.text("Lūdzu pierakstieties, lai turpinātu tīmekļa vietnes darbību.")  
    st.button("Pierakstieties ar Google", on_click=st.login, icon="🔒")
```

#### 5.2. koda fragments. Pierakstīšanās lapa

Pēc veiksmīgas Google konta pierakstīšanās *st.user* objekts tiek iestatīts ar lietotāja Google konta datiem. Nākamajā solī Streamlit ietvars pārbauda vai lietotāja WebODM projekta ID nav iestatīts sīkdatnē. Ja sīkdatne nesatur lietotāja WebODM projekta ID, Streamlit ietvars meklē PostgreSQL datubāzē eksistējošu lietotāju pēc Google konta e-pasta adreses. Gadījumā, ja lietotājs jau eksistē *lietotāji* datubāzes tabulā, lietotāja WebODM projekta ID vērtība tiek iestatīti sīkdatnē un sistēmas darbība turpinās.

Bet, ja datubāze nesatur pašreizējo lietotāju, tas nozīmē, ka lietotājs pirmo reizi pierakstās sistēmā vai nav pilnvarots. Lai noskaidrotu lietotāja e-pasta autorizācijas statusu, Streamlit meklē, vai lietotāja e-pasts ir datubāzes *autorizeti\_epasti* tabulā. Nepilnvarotas e-pasta adreses pierakstīšanās process sistēmā tiek noraidīts un sistēma paziņo, lai lietotājs sazinās ar tīmekļa vietnes administratoru. Pilnvarotam jaunam lietotājam sistēma izveido WebODM projektu caur API pieprasījumu, kuru rezultāts – projekta ID – tiek saglabāts datubāzē tabulā *lietotaji*, kas savieno lietotāja e-pastu ar attiecīgo WebODM projektu. Pēc datubāzes lietotāja izveides, Streamlit ietvars saglabā jauni veidoto projekta ID sīkdatnē.



5.3. att. Lietotāja autorizācijas blokshēma

Lietotājs spēj izrakstīties no tīmekļa vietnes, izmantojot pogu “Izrakstīties”. Izrakstīšanās procesa sākumā visi dati, kas ir saistīti ar lietotāja sesiju, tiek izdzēsti no sīkdatnes, un, izmantojot Streamlit ietvara *st.logout* funkcionalitāti, lietotājs tiek izrakstīts no sistēmas.

#### 4.4. Ortofoto izveides sistēma

Ortofoto izveide ir pieejama tīmekļa vietnes mājas lapā vai caur sāna navigācijas joslu, spiežot uz “Ortofoto izveide” pogu zem “Ortofoto” nodaļas. Navigācijas josla tiek veidota caur Streamlit ietvara iebūvēto *st.navigation* funkciju, kas ļauj izveidot vairāku lapu tīmekļa vietni. Koda fragmentā 5.3. tiek demonstrēts, ka katra tīmekļa vietnes lapa ir *st.Page* objekts, kurš tiek asociēts ar kādu Streamlit Python failu. Visus *st.Page* objektus apvieno navigācijas joslā *st.navigation* izveidotā vārdnīcā, kur katra vārdnīcas atslēga ir navigācijas joslas kategorijas nosaukums un vērtība ir masīvs, kas satur kategorijas lapas objektus.

```

ortofoto_izveide = st.Page("lapas/ortofoto_izveide.py", title="Ortofoto izveide", icon="↗")
ortofoto_parvalde = st.Page("lapas/ortofoto_parvalde.py", title="Mani ortofoto", icon="🖼")
sensoru_dati = st.Page("lapas/sensoru_dati.py", title="Sensoru dati",

```

```

    icon="🔗")
pg = st.navigation({"Ortofoto": [ortofoto_izveide, ortofoto_parvalde], "Sensora
    Dati": [sensoru_dati]})

pg.run()

```

### 5.3. koda fragments. Navigācijas joslas konfigurācija

Bezpilota gaisa kuģu bildes ortofoto izveidei, lietotājam ir nepieciešams nodrošināt sistēmai sekojošos datus: ortofoto/kartes nosaukumu, ortofoto izšķirtspēju, izvēloties vērtību skalā no 0.0 līdz 5.0, ortofoto attēlu uzņemšanas datumu un ZIP failu, kas satur bezpilota gaisa kuģu attēlus. Dati tiek iegūti to lietotāja, izmantojot attiecīgos Streamlit ietvara datu ievades elementus: *st.text\_input*, *st.slider*, *st.date\_input* un *st.file\_uploader*. Kartes nosaukums ir primārais indikators, caur kuru lietotājs atpazīs pašu veidotās ortofoto kartes “Mani ortofoto” lapā. Datu integratītās nolūkiem nosaukumam ir 100 rakstu zīmju garuma ierobežojums, kuru lietotājs nespēj pārsniegt. Ortofoto izšķirtspēja ir saistīta ar WebODM apstrādes *orthophoto-resolution* opciju, kas nozīmē ortofoto izšķirtspēja cm/piksēlis. Izšķirtspēja noklusējumā ir uzstādīta uz 3.0 no 5.0, lai taupītu WebODM servera resursus. Ortofoto datums ir datums, kad bezpilota gaisa kuģu attēli tika uzņemti un datums, ko sistēma izmantos temperatūras un mitruma sensora datu pieprasījumos. Streamlit *st.date\_input* datu ievades elementa, kā arī citu vizuālo komponenšu CSS tika pārveidots, lai, piemēram, iztulkotu datuma izvēlnes mēnešu nosaukumus latviešu valodā. Lietotājs augšupielādē ZIP failu, kas satur bezpilota gaisa kuļa attēlus un kura izmērs nevar pārsniegt 400 MB. Sistēmā ir iespējams konfigurēt vēl lielāku ierobežojumu, bet, lai taupītu tīmekļa vietnes servera resursus un ierobežotu ļaunprātīga lietotāja sistēmas ļaunprātīgu izmantošanu, tika ieviests ierobežojums. Darba autors uzstādīja šādu ierobežojumu, balstoties uz to, ka faktiski lauka attēli, ko darba autors izmantoja sistēmas izstrādes procesā, aizņēma aptuveni 65 MB.

Kad lietotāja izvēlētais ZIP fails ir pabeidzis augšupielādes procesu uz Streamlit servera, sistēma atver ZIP failu un izmantojot ciklu iziet cauri visiem failiem, kas atrodas ZIP failā, pārbaudot vai fails atbilst derīgam formātam – JPG. Ja fails ir pareizajā formātā, sistēma atver to un ielādē *io* pakotnes *BytesIO* funkcijā, kas izveido atmiņā esošu faila līdzīgu objektu. Izmantojot šo objektu, sistēmu uzstāda attiecīgu faila nosaukumu (*.name* īpašību) un beigās pievieno failu *atteli* sarakstam, kurš satur visus lietotāja augšupielādētos derīgos attēlus. Ja lietotājs ir pilnīgi ievadījis visus datus, poga “Ģenerēt karti” kļūst aktīva, un, to nospiežot, tiek izsaukta funkcija *generet\_karti*.

Funkcija *generet\_karti* uzsāk ortofoto izveides procesu (5.4. att.), nosūtot POST pieprasījumu uz WebODM *tasks* galapunktu un izveidojot jaunu WebODM uzdevumu (task) ar nosacītiem datu (data) parametriem: opcijas (options), kuras norāda WebODM API ar kādām opcijām apstrādāt ortofoto, nosaukumu (name), kas tiek iestatīts kā uzdevuma nosaukums un daļējs (partial) būla vērtība, kura pasaka WebODM API, lai nesāk uzdevuma apstrādi uzreiz.

Uzdevuma iestatījumu vērtības, kuras tiek izmantotas ortofoto apstrādē iekļauj:

- Ortofoto izšķirtspēja (*orthophoto-resolution*) – vienīgais dinamiskais iestatījums, kuru vērtību lietotājs tieši uzstāda Streamlit ietvara ortofoto izstrādes procesā. Vērtība nosaka ortofoto izšķirtspēju mēriņumā – cm/pikselis. Šis ir primārais iestatījums, kas ietekmē apstrādes procesa ilgumu un kvalitāti.
- Struktūra no kustības algoritms (*sfm-algorithm*) – tiek izmantota “planar” vērtība, jo plakņu ainām, kas uzņemtas fiksētā augstumā ar attēliem, kuros redzami tikai nadīri punkti (punkti, kas ir tieši zem bezpilota gaisa kuģa), apstrāde ar “planar” argumentu var būt daudz ātrāka. [14]
- Ātrs ortofoto (*fast-orthophoto*) – iestatījuma vērtība ir uzstādīta uz “True”, jo ortofoto apstrādes procesā nav nepieciešama 3D modeļa izveide, tikai ortofoto.
- Saskaņot kaimiņus (*matcher-neighbors*) – veic attēlu salīdzināšanu ar tuvākajiem attēliem, pamatojoties uz GPS exif datiem. Šī ir veikspējas optimizācija, jo tā samazina kopējo funkciju salīdzinājumu skaitu. WebODM iesaka izmantot “4” vērtību lauka ortofoto izveidei. Tā kā bezpilota gaisa kuģu attēliem jau ir liels pārklājuma līmenis (85%), saskaņošana ar vairāk nekā 4 kaimiņiem bieži vien nesniedz pievienotu kvalitāti, tā tikai palēnina procesu.
- Flīzes (*tiles*) – opcija vērtība ir uzstādīta uz *True* un ġenerē statiskas flīzes ortofoto bildēm un digitālu augstuma modeli (DEM), kas ir piemērotas kartēšanas pakotnei kā Leaflet, kas tiek izmantots Streamlit ietvara kartes vizualizēšanai. Flīzes uzlabo kartes skata lietojamību un veikspēju, jo tiek pieprasīti tikai ortofoto dati, kas tiks vizualizēti.

Visi pārējie WebODM apstrādes iestatījumi izmantoja noklusējuma vērtības un netika mainītas. Ja WebODM uzdevums tiek veiksmīgi izveidots, sistēma iestata uzdevuma datus Streamlit sesijas atmiņā vēlākai lietošanai un izveido datubāzes *odm\_uzdevumi* tabulā jaunu ierakstu, saglabājot uzdevuma ID kopā ar lietotāja izvēlēto datumu, kuru nav iespējams saglabāt paša WebODM uzdevuma objektā.

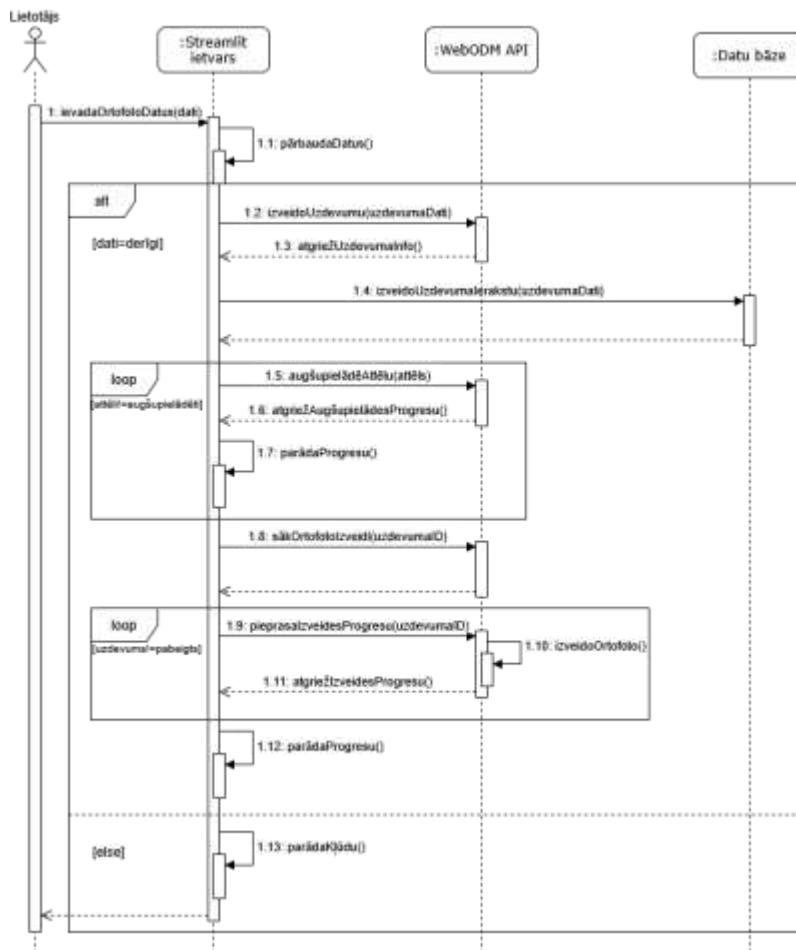
Pirms uzdevuma apstrādes ir nepieciešams augšupielādēt visus attēlus WebODM sistēmā, bet failu apjoms un izmērs ir pārāk liels, lai to veiktu vienā pieprasījumā, tāpēc attēli tiek augšupielādēti pa vienam, izpildot iterāciju ar *for* ciklu, kur katrā iterācija atbilst unikālam failam. Augšupielādes progress tiek aprēķināts izdalot iterācijas indeksa vērtību ar kopējo failu skaitu mīnus viens priekš indeksa saskaņošanas. Sistēma vizualizē progresu caur Streamlit ietvara *st.progress* progresu joslas elementu. Lietotājam ir iespējams atceļt augšupielādi, kā arī ortofoto apstrādi, kas izdzēš WebODM uzdevumu un datubāzes uzdevuma ierakstu. Katrā attēla faila nosaukums, dati baitos un faila tips tiek iestatīti slēgtajā sarakstā, kurš tiek sūtīts caur POST pieprasījuma *failes* atribūtu (5.4. koda fragments). Kad visi attēli ir augšupielādēti WebODM uzdevumā, ortofoto apstrāde tiek uzsākta, sūtot pieprasījumu uz *commit* galapunktu.

```
for indeks, fails in enumerate(st.session_state.faili):
    progress = indeks / (failu_skaits - 1) # Tieki aprēķināts progress
    progesa_josla.progress(progress, progesa_text)

    atteli = [("images", (fails.name, fails, "image/jpg"))]
    augsupieladet_odm_attelus_pec_id(st.session_state.uzdevuma_id, atteli)
else: # Kad visi faili ir augšupielādēti, tiek uzsākta ortofoto apstrāde
    sakt_uzdevumu_pec_id(st.session_state.uzdevuma_id)
    st.session_state.uzdevums_augsupielade = False
    st.session_state.uzdevums_aktivs = True
    st.rerun()
```

#### 5.4. koda fragments. Attēlu augšupielādes cikls

Ortofoto apstrādē, līdzīgi kā failu augšupielādē, procesa progress tiek vizualizēts ar *st.progress* elementu, kuru vērtība tiek iegūta no WebODM uzdevuma informācijas *running\_progress* vērtības. Progress tiek pārbaudīts *while* ciklā 5 sekunžu intervālā līdz WebODM uzdevuma vērtība *status* ir vienāds ar 40 – apstrāde ir pabeigta. Pēc apstrādes, lietotājs tiek paziņots par veiksmīgo ortofoto izveidi un ka ir iespējams apskatīt to dziļāk “Mani ortofoto” tīmekļa lapā.



5.4. att. WebODM uzdevuma izveides secības diagramma

#### 4.5. Lietotāju ortofoto kartes pārvaldes sistēma

Lietotāju veidotās ortofoto kartes tiek glabātas WebODM sistēmā un, lai lietotājs spētu piekļūt pie tām, ir nepieciešams veidot lietotāju saskarni, kas savieno Streamlit ietvaru ar WebODM lietotāja datiem. Šim nolūkam tika veidota “Mani ortofoto” tīmekļa lapa, kur lietotājs spēj apskatīt sarakstā visas savas ortofoto kartes un veikt vairākas darbības – atvērt ortofoto kartes skatā, lejupielādēt ortofoto GeoTIFF formātā un izdzēst ortofoto karti. Lietotājs spēj atjaunot ortofoto sarakstu un atvērt ortofoto kartes skatu no GeoTIFF failu, bet operācijas ar GeoTIFF failu ir ierobežotas, jo kartei nav pieejamas WebODM ortofoto flīzes, kuras padara operācijas ar karti ievērojami ātrākas, datubāzē netiek saglabātas sensoru ierīces koordinātas un veģetācijas veselības/zaļuma indeksa karte (VARI), ko WebODM izveido, nav pieejama. Bet, ja lietotājs tomēr grib atvērt GeoTIFF failu karte skatā, process tiek uzsākts, izmantojot “Atvērt GeoTIFF” pogu, iestatot ortofoto datumu un izvēloties GeoTIFF failu caur faila pārlūku.

Ja Streamlit sesijas atmiņā jau neeksistē *odm\_uzdevumi* mainīgais, Streamlit ietvars pieprasī visus lietotāja uzdevumus (tasks) no WebODM API caur GET pieprasījumu uz

lietotāja projekta *tasks* galapunktu, saglabājot uzdevumus Streamlit sesijas atmiņas sarakstā. Individuāli uzdevumi tiek attēloti Streamlit *st.container* konteinerā elementā ar iespējotu apmales attēlošanu, tādējādi katrs uzdevums tiek vizuāli atdalīts atsevišķā elementā.

Lietotājs spēj pārvaldīt savus veidotos WebODM uzdevumus caur sarakstu. Sarakstā tiek attēloti WebODM uzdevuma dati – izveides datums un laiks, nosaukums un izšķirtspējas kvalitātes vērtība. Katram izpildītam WebODM uzdevumam ir pieejamas divas darbības:

1. Spiežot uz kartes ikonu, sensora dati tiek ielādēti no sensora datu API, izmantojot izvēlēto ortofoto datumu, un ortofoto tiek vizualizēts kartes skatā.
2. Spiežot uz faila ikonu, Streamlit ietvars, izmantojot funkciju *lejupladet\_tif\_pec\_id*, lejupielādē GeoTIFF ortofoto failu no WebODM API *download/orthophoto.tif* galapunktu un lietotājs to var saglabāt lokāli savā datorā, izmantojot *st.download\_button* elementu.

Nepabeigliem WebODM uzdevumiem lietotājs nespēj veikt iepriekš pieminētās operācijas un operāciju pogas ir atspējotas.

WebODM ortofoto uzdevumu un datubāzes ortofoto ieraksta dzēšana ir iespējama ar atkritumu tvertnes pogas spiedienu. Spiediens veic *DELETE* datubāzes vaicājumu pēc uzdevuma ID un POST pieprasījumu uz WebODM API uzdevuma *remove* galapunktu tādejādi, izdēšot ortofoto datus no abām glabāšanas sistēmām.

#### 4.6. Sensoru datu iestatīšana

Galvenā tīmekļa vietnes daļa, kur lietotājs pavada visvairāk laiku, apskatot un analizējot lauka augu veselību, krāsu un temperatūras vai mitruma sensoru datu pārklājumus, ir ortofoto kartes skats. Kartes skatu var sadalīt divās daļās: ortofoto un VARI (ja ortofoto apstrāde ir veikta caur tīmekļa vietni) daļā un sensoru datu daļā. Kartes skatam lietotājs var piekļūt, atverot ortofoto GeoTIFF failu, izmantojot "Atvērt GeoTIFF" pogu, vai izvēloties kādu ortofoto saraksta WebODM uzdevumiem caur kartes ikonas pogu. Kad uzdevuma kartes ikonas poga tiek nospiesta, sistēma pieprasī WebODM uzdevuma informāciju ar API pieprasījumu un veic datubāzes vaicājumu uz *odm\_uzdevumi* tabulu, lai iegūtu uzdevuma iestatīto datumu. Abus iegūtos datus – WebODM uzdevuma informāciju un datubāzes ortofoto datumu – sistēma saglabā Streamlit sesijas atmiņā, kas sistēmai signalizē parādīt kartes skatu.

Kad lietotājs nonāk kartes skatā, sistēma, ja Streamlit sesijas atmiņā nepastāv patiesa *ortofoto\_sensora\_dati* vērtība, mēģina pieprasīt sensora datus ar API pieprasījumu, izmantojot *dabut\_visus\_sensora\_iеракстus* funkciju un datumu argumentu, kas tiek aprēķināts, izvēlētajam ortofoto datumam pieskaitot vienu dienu. Datumu diapazona tiek

formatēta pareizajā Oracle sensora datu API formātā un sistēma veic sensora datu pieprasījumu. Visi iegūtie sensora dati tiek pievienoti *visi\_iеракsti* sarakstam, kas satur visus sensora datu ierakstus. Izmantojot *while* ciklu un pārbaudot *hasMore* pieprasījuma būla vērtību, sistēma pārbauda, vai ir pieejami papildu sensora datu ieraksti. Cikls turpinās, līdz visi sensora datu ieraksti, no pieprasītās datuma diapazonas ir pievienoti sarakstam un *hasMore* vērtība kļūst nepatiesu (5.5. koda fragments).

```

sensora_dati = dabut_sensora_datus(st.secrets.sensoru_datu_url.format(
    str_diapzona[0], str_diapzona[1]))

visi_iераксти += sensora_dati["items"] #Sensora dati tiek pievienoti sarakstam

while sensora_dati["hasMore"]:#Cikls darbojas, līdz visi dati tiek iegūti
    jaunais_datu_url = [link for link in sensora_dati["links"] if "next" in
link["rel"]]] #Iegūst nākamo pieprasījuma saites objektu

    sensora_dati = dabut_sensora_datus(jaunais_datu_url[0]["href"])

    if sensora_dati:
        visi_iераксти += sensora_dati["items"]

```

#### *5.5. koda fragments. Sensoru datu pieprasīšanas cikls*

Sensora dati tiek atgriezti, bet, ja sensora datu ieraksta sarakstā nav neviens ieraksts, sistēma paziņo lietotājam par sensoru datu trūkumu izvēlētajā diapazonā.

Ja atgriezto sensoru datu saraksts nav tukšs, Streamlit ietvars mēģina apstrādāt sensora datu ierakstus, izsaucot *izveidot\_sensora\_ierices* funkciju, kuras rezultāti tiek saglabāti kešatmiņā. Funkcija izveido un atgriež pirmo laika vērtību, kurā eksistē datu ieraksts, kā arī vārdnīcu – *sensora\_ierices*, kur katras atslēga ir unikāls sensora ierīces ID un tās vērtības ir sensora koordināta vērtība – *koordinatas* un *dati* saraksts, kas satur visus attiecīgā sensora mērījumu ierakstus. Ja ortofoto ir atvērts caur lietotāja WebODM uzdevumu sarakstu, sistēma pieprasa sensoru koordinātas no datubāzes pēc uzdevuma ID. Gadījumā, ja sensora koordinātas izvēlētajam WebODM uzdevumam eksistē datubāzē, koordināta tiek iestatīta sesijas atmiņas vārdnīcā *sensora\_ierices* kā attiecīgās sensora ierīces *koordinatas* vērtība. Bet, ja sensora koordinātas WebODM uzdevumam neeksistē datubāzē, tiek veidoti jauni *sensoru\_koordinatas* tabulas ieraksti, kuru ģeogrāfiskais garums un platumis ir uzstādīts uz *NULL*.

## **4.7. Sensoru datu un ortofoto vizualizēšana GIS vidē**

Pēc ortofoto datu ielādēšanas, lietotājs spēj pārvietoties lietotāja saskarnē starp divām cilnēm – kartes skata cilni un sensoru datu diagrammas cilni.

Kartes skata cilnē ortofoto sensoru datuma un laika kontroles elementi atrodas virs kartes, šie elementi nosaka, kuri dati tiek pieprasīti un vizualizēti. Ortofoto uzdevuma datums nosaka sensoru datu datuma diapazonas sākumu un laiks nosaka tieši kuri individuālie sensora datu ieraksti tiek vizualizēti GIS vidē. Kad lietotājs izmaina ortofoto datumu, sistēma atiestata sensoru ierīces sarakstu, jo ir nepieciešama jauna sensoru datu uzstādīšana, un, ja atvērtais ortofoto ir apstrādāts caur WebODM, sistēma datubāzē nomaina datumu attiecīgajam WebODM uzdevumam, kā arī izdzēš visas vecās sensoru ierīces koordinātas no datubāzes.

Galvenā kartes skata komponente – paša karte, tiek veidota, izsaucot *izveidot\_karti* funkciju. Šīs funkcijas atgriešanas vērtība – Folium pakotnes karte tiek glabāta kešatmiņā caur *@st.cache\_data* dekoratora pielietojumu, kas nodrošina, ka karte tikai tiek pārtaisīta, kad izmainās kartes izveidei kritiskas vērtības.

Lai izveidotu Folium karti – *folium.Map*, sistēmas pirmā operācija ir aprēķināt kartes sākuma punktu (ortofoto centru). Ortofoto centra aprēķinam (5.6. koda fragments) ir nepieciešams iegūt ortofoto koordinātu robežas, kuras WebODM uzdevumam jau ir pieejamas uzdevuma datu *extent* vērtībā, bet, ja ortofoto tiek ielādēts no GeoTIFF faila, centra koordinātu, datu saiti, kura satur GeoTIFF faila ortofoto PNG faila formātā, un Folium kartes robežas dati tiek aprēķināti un atgriezti, izsaucot funkciju *ieladet\_tif\_datus*.

```
@st.cache_data
def izrekinat_ortofoto_centru(robeza):
    min_lon, min_lat, max_lon, max_lat = robeza #Sadala robežas koordinātas

    # Aprēķināt attēla centru izmantojot ģeogrāfisko garumu un platumu
    centra_lon = (min_lon + max_lon) / 2
    centra_lat = (min_lat + max_lat) / 2

    return centra_lat, centra_lon
```

#### 5.6. koda fragments. Ortofoto centra aprēķinu funkcija

Pēc centra aprēķināšanas, tiek izveidota Folium karte – *folium.Map*, izmantojot iegūto centra koordinātas, tālummaiņas līmeņa maksimālo, minimālo un sākuma vērtībām. Visi ortofoto, VARI un sensora datu slāni tiks pievienoti izveidotajai Folium kartei.

Folium kartei tiek pievienots ESRI pasaules pamat kartes satelīta flīzes slānis (5.7. koda fragments), kuru Folium karte pieprasa, izmantojot specifiskas X, Y un Z koordinātas,

lai iegūtu tikai nepieciešamos satelīta kartes datus. Kartes slānis attēlo apkārtni kā satelītattēlus.

```
folium.TileLayer(
    tiles='https://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServer/tile/{z}/{y}/{x}',
    attr="Tiles © Esri – Source: Esri, i-cubed, USDA, USGS, AEX, GeoEye, Getmapping, Aerogrid, IGN, IGP, UPR-EGP, and the GIS User Community",
    name="Satelīta flīzes",
    overlay=False,           # Neatļauj lietotājam kontrolēt slāņa redzamību
    zoom_start=ZOOM_START, # Tuvināšanas līmeņa iestatīšana
    max_zoom=MAX_ZOOM,
    min_zoom=MIN_ZOOM,
    max_native_zoom= 20,    # Slānis nodrošina tikai tuvināšanu līdz 20.
    show=True,              # Ir pamata slānis tāpēc vienmēr tiek parādīts
    z_index=1,              # Norāda slāņa vizualizācijas prioritāti
).add_to(m)             # Slānis tiek pievienots kartei - m
```

### 5.7. koda fragments. Satelīta flīzes slānis

#### 4.7.1 Ortofoto vizualizācija GIS vidē:

1. Gadījumā, ja lietotājs izvēlās atvērt ortofoto no GeoTIFF faila: Funkcija *ielađet\_tif\_datus* atver un iegūst GeoTIFF faila datus, izsaucot *rasterio.open* metodi, un pārveido GeoTIFF faila koordinātas atskaites sistēmu (CRS) uz EPSG:4326, ko Folium kartēšanas pakotne izmanto, lai spētu aprēķināt ortofoto robežas attēla vizualizācijai. Funkcija arī iegūst GeoTIFF attēlu RGB masīvu, pievieno alfa kanālu, lai aizvietotu ortofoto tukšos datus ar caurspīdīgu apgabalu. Karte attēlo GeoTIFF ortofoto kā Folium *ImageOverlay* slāni, izmantojot aprēķinātās ortofoto robežas un datu saiti, kura norāda uz ortofoto PNG datiem atmiņā. *ImageOverlay* slānis tiek pievienots izveidotajai Folium kartei (*m*) kā vienīgais noklusējuma ortofoto slānis.
2. Gadījumā, ja lietotājs izvēlās kādu no WebODM uzdevumiem no ortofoto saraksta: Nav nepieciešams izsaukt funkciju *ielađet\_tif\_datus*, jo ortofoto dati, konkrētāk – ortofoto flīzes dati, tiek mājoti WebODM serverī, kurus sistēma izmantos ortofoto vizualizācijai. Tas nozīmē, ka ortofoto slāni kartē var pievienot, izmantojot Folium *folium.TileLayer* funkciju, kā flīzes slāni. Līdzīgi kā satelīta pamata kartes flīzes slānis, Folium ortofoto flīzes slānis pieprasīta tikai nepieciešamos datus konkrētā koordinātā. Ortofoto flīzes URL – *ortofoto\_flīzes\_url* tiek uzstādīts, atsaucoties uz WebODM uzdevuma *orthophoto/tiles/{z}/{x}/{y}* galapunktu, pievienojot WebODM JWT autorizācijas žetonu galapunkta beigās kā *?jwt* URL vaicājumu (query), autorizācijas nolūkiem.

Līdzīgi tiek pievienots augu zaļuma indeksa (VARI) flīzes slānis, apvienojot iepriekš iegūto ortofoto flīzes URL ar virkni –

```
"&formula=VARI&bands=auto&color_map=rdylgn&rescale=0.19014084507042253,0.19827586206896552". Šīs WebODM URL vaicājumu opcijas konfigurē dinamisko VARI izveidi, aprēķinot VARI veģetācijas indeksu, izmantojot automātiski atlasītas attēla joslas, vizualizējot rezultātu ar sarkanu-dzeltenu-zaļu krāsu skalu un mērogojot krāsu kartējumu, lai attēlotu indeksa vērtības noteiktā skaitliskā diapazonā. Kartes izveides funkcijas beigās, abi flīzes slāņi tiek pievienoti Folium elementam – folium.plugins.GroupedLayerControl, kas izveido flīzes slāņa kontroles lietotājas saskarnes elementu un satur ekskluzīvas pievienoto slāņu kontroles operācijas.
```

#### 4.7.2 Sensoru datu integrācija GIS vidē

Sistēma funkcijā *izveidot\_karti* izfiltrē visas sensora ierīces ar uzstādītām koordinātām, saglabājot tās vārdnīcā sensora\_ierices\_ar\_koord. Visas sensoru ierīces ar koordinātām tiek attēlotas kartē kā caurspīdīgi apli ar zilu apmali, uz kuru uzspiežot, tiek attēlots sensora ID un koordinātu vērtība. Cikliski ejot cauri vārdnīcai *sensora\_ierices\_ar\_koord*, sistēma mēģina atrast katram sensoram datu ierakstu, kura laika vērtība sakrīt ar lietotāja izvēlēto sensoru datu laiku. Ja tiek atrasts šāds ieraksts, sistēma pārbauda katru mērījumu – *air temperature*, *air humidity*, *soil temperature 1*, *soil temperature 2*, *soil moisture 1*, *soil moisture 2* – un izveido vārdnīcu ar tiem mērījumiem, kuru vērtības ir patiesas un nepieciešamas vizualizācijai GIS vidē. Katram patiesam sensoru mērījumam tiek izveidota unikāla *folium.FeatureGroup* elementu grupa, kas tiek saglabāta vārdnīcā, kur atslēgas ir mērījumu nosaukumi, bet vērtības – atbilstošie *folium.FeatureGroup* slāņu objekti. Mērījumu vērtības tiek attēlotas kā *folium.Circle* apļa pārklājuma objekti, kuri tiek pievienoti atbilstošajai mērījumu grupai (5.7. koda fragments).

```
folium.Circle(
    location=sensora_datu_ieraksts["koordinatas"], # Apļa pārklājuma koordināta
    radius=4 if "soil" in merijuma_nosaukums else 5, # Augsnes vērtībām ir
    mazāks rādiuss
    color=robezas_krasa,
    fill=True,
    fill_color=dabut_krasu(merijuma_vertiba, merijuma_nosaukums),
    fill_opacity=0.2, # Aļa pārklājums ir daļēji caurspīdīgs
    popup=f'{datu_nosaukums}: {merijuma_vertiba}',
    weight=1,
    tooltip=vertibas_virkne
```

```
).add_to(slani.get(merijuma_nosaukums)) # Tieki pievienots attiecīgajam slānim
```

### 5.7. koda frgments. Mērijuma datu aplū pārklājums

Apļa pārklājuma krāsa tiek aprēķināta, izsaucot funkciju *dabut\_krasu*, kas izvēlas piemērotu krāsu skalu atkarībā no mērījuma tipa (temperatūra vai mitrums) un pielāgo to attiecīgajam vērtību diapazonam. Tā atgriež krāsu, kas atbilst konkrētajai mērījumu vērtībai šajā skalā. Apļa pārklājuma *popup* atribūta vērtība vizualizē sensora mērījuma vērtību un tiek attēlota, ja lietotājs uzspiež uz pārklājumu. Visi sensoru datu slāni tiek pievienoti *folium.plugins.GroupedLayerControl* grupai, caur kuru lietotājs spēj ieslēgt vai izslēgt atsevišķus datu slāņus.

#### 4.7.3 Sensoru koordinātu uzstādīšana

Lai sistēma spētu vizualizēt sensoru noteiktā koordinātē, lietotājam ir jāatlasa un jāiestata koordinātas sistēmā. Visas koordinātu darbības – uzstādīšana, rediģēšana un dzēšana – ir pieejamas zem kartes skata. Sistēma sadala sensoru ierīces divās grupās: sensori ar uzstādītām koordinātām un sensori bez tām. Caur nolaižamas izvēlnes lietotājs spēj izvēlēties sensoru pēc ID, kuram veikt koordinātu darbību. Koordinātu iestatīšanas gadījumā tiek piedāvāti tikai tie sensori, kuriem nav iestatītas koordinātas, savukārt sensoriem ar jau iestatītām koordinātām ir iespējams veikt koordinātu labošanu vai dzēšanu. Kad lietotājs ir izvēlējies sensoru un vēlamo darbību, sistēma iestata sesijas atmiņas mainīgā *spiediena\_rezims* vērtību uz patiesu, tādējādi aktivizējot režīmu, kurā sistēma sāk uztvert kartē pēdējo nospiesto punktu koordinātu un saglabā to sesijas atmiņā mainīgajā – *izveleta\_koordinate*. Jebkura koordinātu izmaiņas operācija saglabā jauno koordinātu vērtību (*izveleta\_koordinate*) gan lokāli, rediģējot sesijas atmiņas vārdnīcu *sensora\_iерices*, gan tīmeklī, atjauninot datubāzes tabulas *sensora\_iерices* ierakstu pēc uzdevuma un sensora ID. Koordinātas datubāzē tiek saglabātas tikai tad, ja lietotājs tās uzstāda no WebODM izveidota ortofoto, kas tika izvēlēts no ortofoto saraksta, nevis caur GeoTIFF faila augšupielādi.

### 4.8. Sensoru datu diagramma

Sensoru datu diagrammas ir pieejama tīmekļa vietnes “Sensoru dati” lapā un kartes skata “Sensoru dati” cilnē. Lietotājs “Sensoru dati” lapā, izmantojot *st.date\_input* datumu izvēles elementu, spēj izvēlēties datumu diapazonu no kurās sistēma pieprasīs visus sensora datus, savukārt diagramma kartes skata cilnē attēlos tikai tos sensora datus, kas atbilst lietotāja izvēlētajai ortofoto datumu vērtībai. Pēc sensoru datu uzstādīšanas, dati tiek padoti kā argumenti *zimet\_sensora\_datus* funkcijai (5.8. koda frgments).

Funkcija uzstāda sensora datus kā *pandas* pakotnes datu rāmi (*pd.DataFrame*), izsaucot *iestatit\_df* funkciju, kas iztulko 2 dimensiju datu rāmju kolonas latviešu valodā un saglabā iegūto rezultātu kešatmiņā. Atpakaļ *zimet\_sensora\_datus* funkcijā sistēma veido nolaižamu izvēlni, kas satur visus iespējamos mērījumu veidus. Atkarībā no izvēlētā mērījumu veida tiek izveidots *altair* pakotnes diagramma, kurā datu rāmis attēlo mērījumu laika gaitā: X ass attēlo laiku, bet Y ass – izvēlētā mērījuma vērtību. Katrai sensoru ierīcei diagrammā tiek piešķirta unikāla krāsa, un, novietojot cursoru virs kādai no vērtībām, tiek attēlots datu ieraksta datums, sensora ID un mērījuma vērtība.

```
def zimet_sensora_datus(sensora_dati):
    df = iestatit_df(sensora_dati)
    izveleta_opcija = st.selectbox("Datu kategorijas: ",list(opcijas.values()))
    diagramma = (
        alt.Chart(df) # Tieki izmantots iegūtais datu rāmis
        .mark_line(point=True)
        .encode(
            x=alt.X("Datums:T", title="Laiks", axis=alt.Axis(format="%H:%M")),
            y=alt.Y(izveleta_opcija + ":Q", title=izveleta_opcija),
            color=alt.Color("Sensora ID:N", title="Sensoru ID"),
            tooltip=["Datums:T", izveleta_opcija + ":Q", "Sensora ID:N"]
        ).interactive()
    )
    st.altair_chart(diagramma) # Diagramma tiek vizualizēta
```

### 5.8. koda fragments. Sensoru datu apstrāde un attēlošana

Izveidotā diagramma tiek vizualizēta, izsaucot Streamlit ietvara iebūvēto Altair diagrammas vizualizēšanas funkciju – *st.altair\_chart*. Rezultējošā diagramma ir interaktīva un atsaucīga – lietotājs spēj pietuvināt, attālināt un detalizēti izpētīt datus.

## 4.9. Sistēmas prototipa testēšana

Prototips tika testēts no 2 aspektiem – manuāla lietotāju ievada datu testēšana un vides testēšana atšķirīgās operētajā sistēmas un pārlūka programmās.

### 4.9.1 Manuāla ievada datu testēšana

Pirmajā sistēmas darbības testā darba autors pievienoja augšupielādētajam ZIP failam, kas satur bezpilota gaisa kuģu attēlus, failus ar nekorektiem faila paplašinājumiem kā PDF, TIF, SQL utt. (5.4. att.).

|             |                     |          |
|-------------|---------------------|----------|
| fails.pdf   | Firefox PDF Docu... | 91 KB    |
| fails.ase   | ASCI-E File         | 928 KB   |
| fails.tif   | TIF File            | 9 424 KB |
| fails.sql   | SQL Source File     | 1 KB     |
| D8_0720.JPG | JPG File            | 938 KB   |
| D8_0710.JPG | JPG File            | 946 KB   |
| D8_0700.JPG | JPG File            | 933 KB   |
| D8_0690.JPG | JPG File            | 924 KB   |
| D8_0680.JPG | JPG File            | 918 KB   |
| D8_0670.JPG | JPG File            | 915 KB   |
| D8_0660.JPG | JPG File            | 887 KB   |
| D8_0650.JPG | JPG File            | 894 KB   |

5.4. att. Nekorektie faila paplašinājumi

Testa rezultātā, izmantojot faila nosaukuma paplašinājuma pārbaudi, sistēma veiksmīgi ignorēja šos klūdainos faila tipus un izveidoja pilnīgu ortofoto rezultātu. Tāpat testos tika novērots, ka gadījumā, ja lietotājs ZIP failā pievieno pareiza formāta JPG failus, kas nav bezpilota gaisa kuģu attēli, sistēma tos ignorē un ortofoto tiek veiksmīgi apstrādāts (5.5. att.), izmantojot tikai derīgos attēlus.



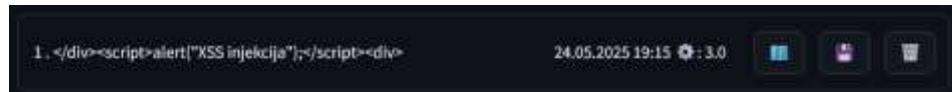
5.5. att. Faila paplašinājuma testa rezultāts

Sistēmā tika testēta lietotāju autorizācijas funkcionalitāte – lietotāji, kuru Google konta e-pasta adreses nav datubāzes pilnvaroto e-pastu tabulā, tiek noliegti no sistēmas funkcionalitātes. Paredzamā sistēmas reakcija ir – noraidīt lietotāju no sistēmas operācijas un attēlot informatīvu ziņu (5.6. att.), par sistēmas noraidīšanas iemeslu. Pēc atkārtotas testēšanas darba autors secina, ka sistēmas darbība atbilst paredzamam rezultātam.



5.6. att. Nepilnvarotas e-pasta adreses paziņojums

Ortofoto izveidē lietotājs ievada kartes nosaukumu teksta ievades elementā. Lai atklātu iespējamu XSS uzbrukumu, ir nepieciešams testēt ievades vērtības integrāciju ar HTML elementiem – vai vērtības tieši tiek injicētas HTML kodā. Izveidotā XSS uzbrukuma vienība – “</div><script>alert("XSS injekcija");</script><div>” mēģina aizvērt DIV bloku, kurā tiek attēlots kartes nosaukums, un injicēt HTML kodā izpildāmu JavaScript kodu. Bet tā kā Streamlit ietvara ievades elementi pirms renderēšanas tiek apstrādāti – pārvērsti uz vienkāršu virkni – XSS uzbrukums nebija veiksmīgs, jo virkne netika izpildīta HTML kodā (5.7. att.).



A screenshot of a Streamlit application window. The top bar shows the URL '127.0.0.1:8501' and the title 'Streamlit Web App'. The main area contains a single line of code: '1. </div><script>alert("XSS injekcija");</script><div>'. Below the code, there is a timestamp '24.05.2025 19:15' and a status indicator '3.0'. On the right side of the window, there are three small icons.

5.7. att. Neveiksmīgs XSS uzbrukums

Ortofoto izveides process tika testēts, vairākas reizes dzēšot attēlus no augšupielādētā ZIP faila bezpilota gaisa kuģu attēlu saraksta. Katram testam tika noņemti vairāki attēli no kopējā saraksta, lai pārbaudītu sistēmas noturību pret nepilnīgu datu ievadi. Rezultātā, pat ja vairāki attēli tiek izņemti ar pietiekami lielu attēlu pārklājumu, ortofoto tiek veiksmīgi izveidots (5.8. att.), izmantojot derīgos attēlu datus.



5.8. att. Attēlu dzēšanas testa rezultāts

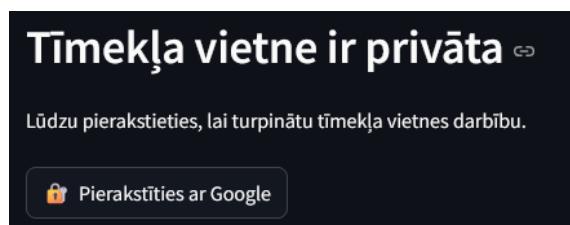
#### 4.9.2 Vides testēšana

Tīmekļa vietne tika veidota un optimizēta, koncentrējoties uz sistēmas datoru vides darbību, sistēmām ar aptuveni 14" lielu displeju. Tīmekļa vietne tika testēta Windows operētāja sistēmas vides Zen, Firefox un Google Chrome interneta pārlūkos, kā arī macOS Monterey operētāja sistēmas vides Google Chrome 112 versijā. Mobilās sistēmas testēšanas rezultāti parādīja, ka tīmekļa vietne darbojās uz Android, un IOS operētājsistēmas ierīcēm. Funkcionalitāte uz visu veida planšetes datoriem un telefoniem ir pilnīga, bet dizaina aspekti nav pilnīgi realizēti – atstarpes starp sistēmas vizuāliem elementiem ne vienmēr ir vienādas vai korekti ierindotas.

## 5. SISTĒMAS DARBĪBAS APRAKSTS UN PĀRSKATS

### 5.1. Lietotāja interfeisa apraksts

Tīmekļa vietnes prototipa lietošana ir tikai paredzēta autorizētiem Google e-pasta kontiem, lai ietaupītu servera resursus un noliegtu neautorizētus vai ļaunprātīgus lietotājus. Neautentificēts lietotājs spēj piekļūt tikai pierakstīšanās lapai (6.1. att.). Izmantojot pogu “Pierakstīties ar Google”, lietotājs sāk pierakstīšanās procesu.



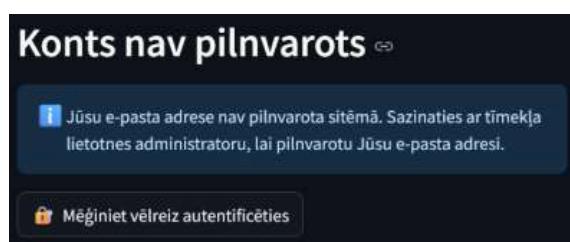
6.1. att. Neautentificēta lietotāja tīmekļa lapa

Līdz ar to, lai sāktu tīmekļa vietnes darbību, lietotājam ir nepieciešams autorizēties sistēmā caur Google konta pierakstīšanos. Lietotājs spēj izvēlēties jau pierakstītu Google kontu vai ieraksta savu Google konta informāciju autentifikācijai (6.2. att.).



6.2. att. Google konta izvēles logs

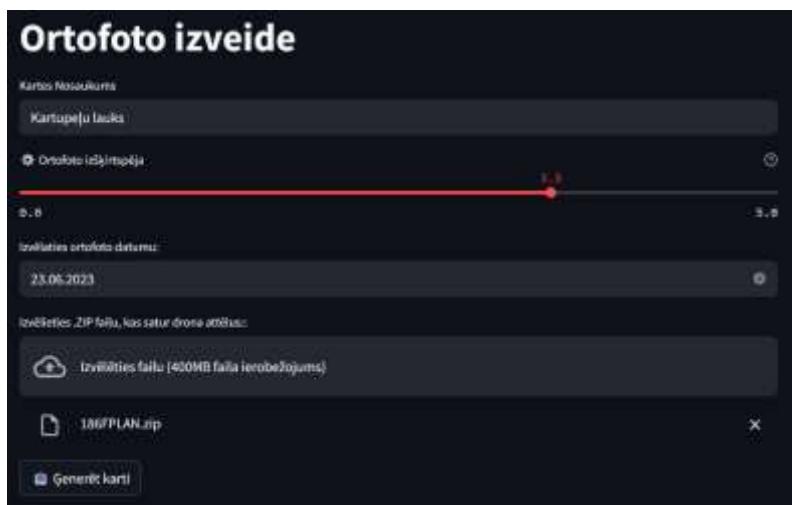
Gadījumā, ja lietotāja e-pasta adrese, kura ir saistīta ar Google kontu, nav datubāzes tabulā *autorizeti\_epasti*, sistēma paziņo par noliegumu (6.3. att.).



6.3. att. Nepilnvarota e-pasta paziņojuma lapa

Ja sistēmas autentifikācija ir paveikta veiksmīgi, lietotājs tiek novirzīts uz “Ortofoto izveide” lapu, kur lietotājs spēj sākt ortofoto izveides procesu (6.4. att.). Ortofoto izveides veidlapā lietotājam ir nepieciešams ierakstīt ortofoto kartes nosaukumu (ne garāks par 100 rakstzīmēm), izvēlēties ortofoto izšķirtspēju skalā no 0.0 līdz 5.0 (kur 5.0 apzīmē maksimālo izšķirtspēju), jānorāda ortofoto izveides datums, no kura tiks ielādēti sensora dati, un

jāaugšupielādē ZIP fails, kas satur visus JPG formāta bezpilota gaisa kuģa attēlus. Kad sistēma konstatē ievada dati ir pareizi, poga “Ģenerēt karti” tiek aktivizēta.



6.4. att. “Ortofoto izveides” tīmekļa lapa

Nospiežot “Ģenerēt karti” pogu, sistēma vispirms augšupielādē visus JPG failus izveidotajā WebODM uzdevumā un pēc tam automātiski uzsāk ortofoto izveides process. Izmantojot “Atcelt kartes izveidi” pogu (6.5. att.), lietotājs spēj apstāt ortofoto izveidi, kas izdzēš uzdevumu no WebODM un PostgreSQL datubāzi. ZIP faila attēlu augšupielādes un ortofoto izveides progress tiek vizualizēts progresā joslā. Lietotājs tiek informēts par veiksmīgu ortofoto izveidi un spēj pāriet uz "Mani ortofoto" lapu, noklikšķinot uz pogu – "Apskatīt izveidoto karti" (6.6. att.).



6.5. att. Ortofoto izveides progresā josla.



6.6. att. Veiksmīgi izveidota ortofoto skats

“Mani ortofoto” tīmekļa lietotnes lapā (6.7. att.) lietotājs spēj apskatīt visus izveidotos ortofoto WebODM uzdevumus un to nosaukumu, izveides datumu un laiku un izšķirtspējas vērtību. “Atjaunināt” poga atjaunina ortofoto sarakstu caur jaunu WebODM API pieprasījumu.



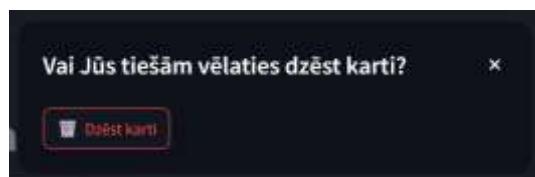
6.7. att. "Mani ortofoto" tīmekļu lapa

Lietotājs var lejupielādēt izveidoto ortofoto karti GeoTIFF formātā, nospiežot disketes ikonas pogu. Pēc pogas nospiešanas, sistēma attēlo uzņirstošu logu, kas attēlo ortofoto nosaukumu un satur lejupielādes pogu (6.8. att.).



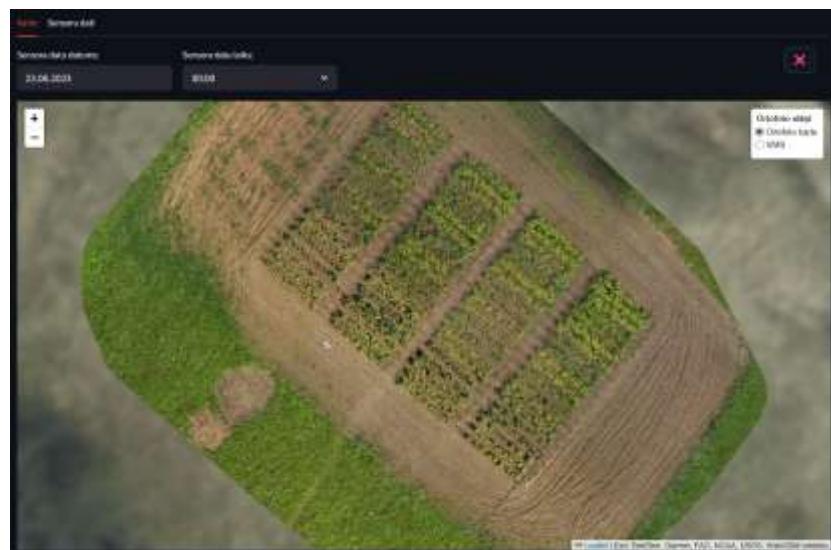
6.8. att. Ortofoto lejupielādes uzņirstošais logs

Līdzīgi, izmantojot atkritumu tvertnes ikonas pogu, lietotājs spēj izdzēst ortofoto no WebODM un datubāzi (6.9. att.).



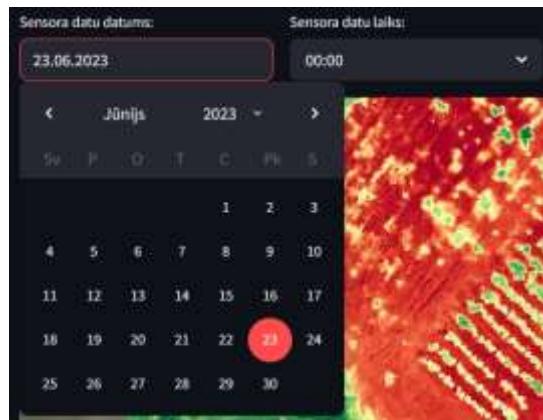
6.9. att. Ortofoto dzēšanas uzņirstošais logs

Lietotājs ir spējīgs nonākt ortofoto kartes skatā (6.10. att.), izmantojot kartes ikonas pogu. Izvēloties ortofoto, automātiski tiek atvērta "Karte" izvēlnes cilne, kurā parādīts centrēts ortofoto ar iespēju to pietuvināt un izpētīt, un tiek ielādēti temperatūras un mitruma sensoru dati.



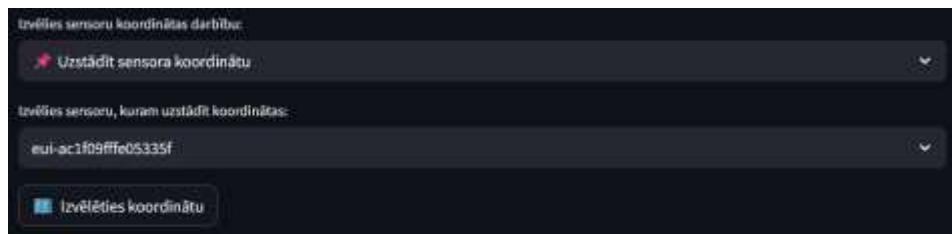
#### 6.10. att. Kartes skats

Kartes skata augšējā kreisajā stūrī ir pieejami sensora datu datuma un laika kontroles elementi (6.11. att.). Izmainot sensora datu datumu, jaunais datums tiek saglabāts datubāzē un sistēma pieprasīs sensoru datus izvēlētajā datumā, kā arī tiek izdzēstas visas iepriekšējās sensoru koordinātas, kuras ir saistītas ar izvēlēto ortofoto.



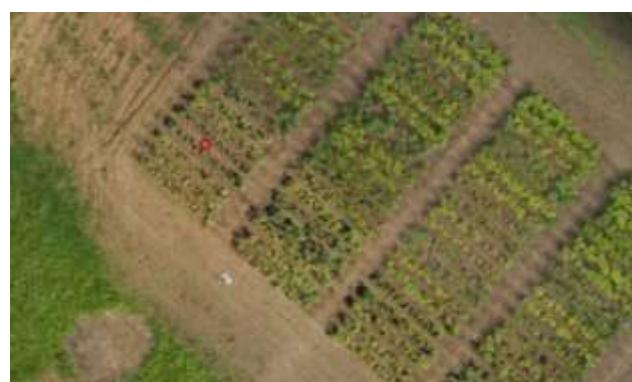
6.11. att. Sensoru datu datuma izvēlne

Zem kartes ir pieejamas sensoru koordinātu operācijas izvēlne (6.12. att.). Lietotājs var izvēlēties koordinātas jebkurai sensora ierīcei bez esošas koordinātes, kā arī mainīt vai dzēst koordinātas jau iepriekš uzstādītai sensora ierīcei. Operācija un sensoru ierīce tiek izvēlēta caur nolaižamās izvēlnes darbību.



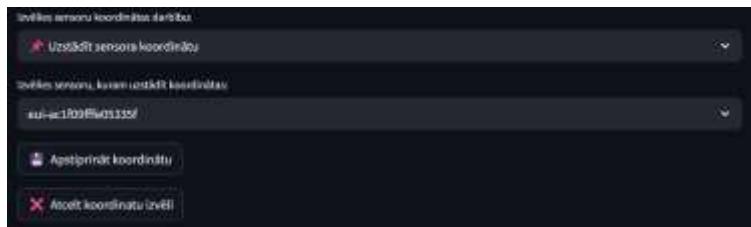
6.12. att. Sensora koordinātu operācijas izvēlne

Kad lietotājs ir izvēlējies operāciju un ierīci, spiežot uz "Izvēlēties koordinātu" pogu, sistēma sāk reģistrēt kartē pēdējo spiesto koordinātu. Izvēlēto koordinātu var atpazīt pēc sarkanā apļa figūras kartē (6.13. att.).



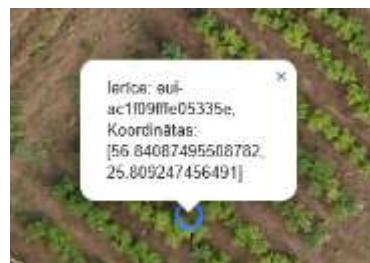
6.13. att. Izvēlēta koordināte kartes skatā

Pēc koordinātu izvēles, poga “Apspirināt koordinātu” tiek aktivizēta (6.14. att.), kuru nospiežot izvēlētā koordināta tiek saistīta un saglabāta ar attiecīgo sensora ierīci. Ja lietotājs vēlas atcelt koordinātu izvēli, to var panākt, izmantojot “Atcelt koordinātu izvēli” pogu.



6.14. att. Koordinātu apspirināšanas izvēlne

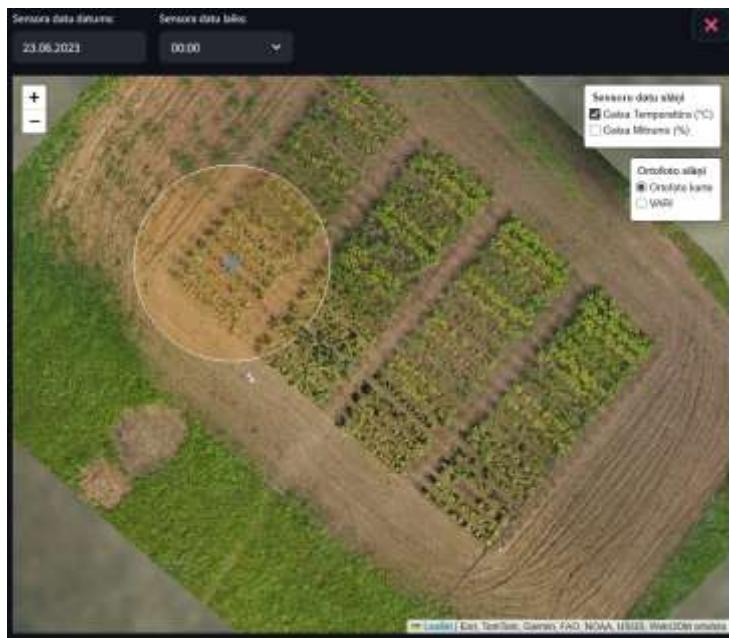
Pēc sensora ierīces koordinātu uzstādīšanas tā tiek attēlota kartē kā zaļš apļa formas pārklājums. Aktivizējot šo pārklājumu ar peles klikšķi, lietotājam tiek attēlota informācija par sensora ierīces ID un koordinātu (6.15. att.).



6.15. att. Sensoru ierīces informācijas uzņirstošais logs

Sistēma automātiski mēģina atrast attiecīgās sensora ierīces mēriju mu ierakstus izvēlētajā sensoru datu laikā. Ja mēriju mu ieraksts tiek atrasts laika punktā, lietotājam mēriju mu dati tiek attēloti kā apļa formas pārklājumu virs kartes (6.16. att.). Visi mēriju mu veidi tiek pievienoti kartes “Sensoru datu slāni” slānu kontrolei kartes augšējā labajā stūrī. Lietotājs spēj iespējot un atspējot datu slānus atkarībā no vizualizācijas nepieciešamības.

Pārklājuma rādiuss un krāsa tiek dinamiski noteikta atkarībā no sensora mēriju mu veida un vērtības. Gaisa mēriju mu tiek piešķirts lielāks vizualizācijas rādiuss nekā augsnēs mēriju mu, lai uzsvērtu to plašāko ietekmes zonu. Pārklājuma krāsa tiek pielāgota mēriju mu veidam – temperatūras mēriju mu tiek izmantota sarkanā krāsas gradācija, savukārt mitruma rādītāji tiek attēloti ar zilās krāsas toni, nodrošinot intuitīvu datu interpretāciju kartes skatā. Krāsas tonis ir atkarīgs no mēriju mu vērtības – temperatūras mēriju mu ar augstāku vērtību tiek attēloti tumšākā sarkanā tonī utt.



6.16. att. Sensoru mērījuma ieraksta vizualizācija kartes skatā

Kad sistēmā pastāv sensora ierīce ar uzstādītu koordinātu, lietotājs ir spējīgs redīgēt eksistējošo koordinātu, izmantojot koordinātu izmaiņas vai dzēšanas funkcionalitāti, kas darbojas līdzīgi koordinātu uzstādīšanas funkcijai (6.17. att.).



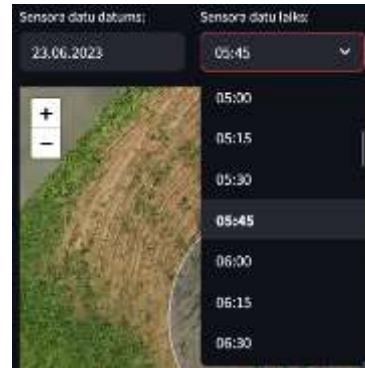
6.17. att. Visas iespējamās koordinātu operācijas

Lietotājam uzspiežot uz pārklājumu, sistēma vizualizē uznirstošu logu (6.18. att.), kas satur mērījuma nosaukuma un vērtības informāciju, savukārt, novietojot kursoru virs pārklājuma, tiek attēlots mērījuma vērtība un mērvienība.



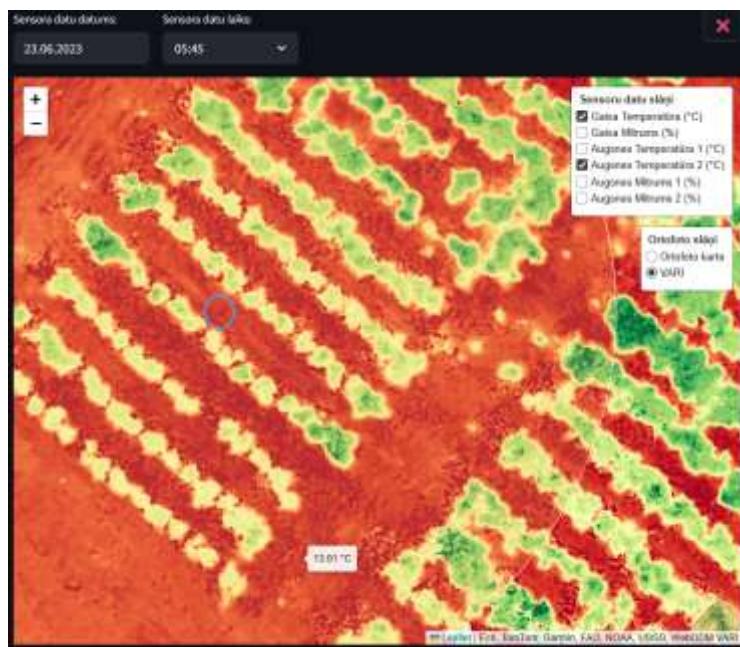
6.18. att. Mērījuma pārklājumu interaktīvā datu vizualizācija

Laika sērijas vērtības izmaiņa notiek, kad lietotājs izvēlās sensoru datu laiku no nolaižamā izvēlnes (6.19. att.). Ja tiek izvēlēts jauna laika vērtība, kas atšķiras no iepriekšējās, sistēma mēģina atrast un vizualizēt sensoru datu mērījumus izvēlētajā laikā.



6.19. att. Sensora datu laika izvēlne

Kā ir redzams pielikumā Nr. 3, lietotājs spēj uzstādīt visas sensoru ierīču koordinātas un analizēt lauka stāvokli, izmantojot vairāku sensoru iekārtu mērījumu pārklājumu slāņus. VARI karte (6.20. att.) ir pieejama, sāna “Ortofoto slāni” kontroles panelī, izvēloties “VARI” opciju. Šis kartes indeksa veids palīdz lauksaimniekiem atrast un identificēt auga apgabalus, kur veģetācijas zaļuma līmenis ir nepietiekamā līmenī. Izmantojot gaisa un augsnes temperatūras un mitruma sensorus lietotājs spēj veikt pilnu analīzi, integrējot visus pieejamos datus kaitīgo faktora izmeklēšanā.



6.20. att. VARI vizualizācija kartes skatā

Ja lietotājs izvēlās GeoTIFF failu ortofoto lauka analīzei, izmantojot “Mani ortofoto” lapas (6.7. att.) “Atvērt GeoTIFF” pogu, lietotājs spēj izvēlēties ortofoto datumu un GeoTIFF failu (6.21. att.), kas tiks izmantots lauka attēlošanā. Tomēr lietotājam ir jāapzinās

sistēmas funkcionalitātes trūkumu GeoTIFF faila darbības gadījumā – sensoru ierīces koordinātas un ortofoto sensoru datums netiek ilgstoši saglabātas datubāzē, VARI nav pieejams, kā arī kartes operācijas ir ievērojami lēnākas.



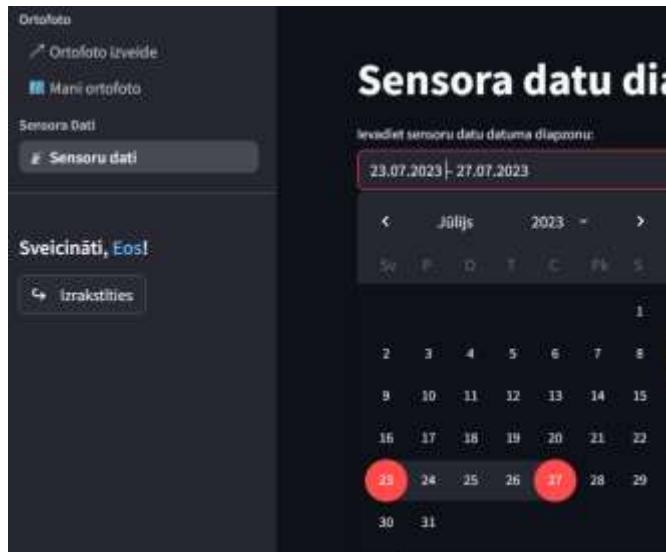
6.21. att. GeoTIFF faila izvēlne

Blakus “Karte” cilnei ortofoto skatā “Sensoru dati” cilnē ir pieejama sensoru datu diagramma (6.22. att.), kas vizualizē lietotāja izvēlēto mērījuma veidu. Lietotājs spēj izvēlēties datu kategoriju no nolaižamās izvēlni un pārskatīt datus, pietuvinot noteiktas datu vienības. Uz Y ass tiek attēlotas lietotāja izvēlētās mērījumu vērtības un uz X – laika sērijas progress. Atšķirīgie sensoru ieraksti tiek izcelti un vizualizēti ar dažādām krāsām, kas atbilst katrai unikālai sensora ierīcei, nodrošinot skaidru un pārskatāmu datu atšķiršanu diagrammas skatā. Kā pielikums Nr. 4 demonstrē, novietojot kurSORU virs mērījumu vienības, tiek attēlots ieraksta laiks, datums, mērījuma vērtība un sensora ID.



6.22. att. Ortofoto datuma sensoru datu diagramma

Tīmekļa vietnes “Sensoru dati” lapā, lietotājs spēj pieprasīt un analizēt sensoru datus specifiskā datuma diapazonā. Izmantojot datu diapazonu no kalendāra izvēlnes (6.23. att.), sistēma pieprasī visus sensora datus no izvēlētās datuma diapazonas. Pielikumā Nr. 5 demonstrē veiksmīgu datuma diapazonas datu vizualizāciju diagrammas skatā.



6.23. att. “Sensoru dati” lapas datuma diapazonas izvēlne

## 5.2. Sistēmas veikspējas un uzturēšanas aspekti

Darba autors, izmantojot tīmekļa pārlūkprogrammas tīkla ātruma ierobežošanas funkcionalitāti, testēja tīmekļa vietnes reakcijas laiku un lietojamību dažādos interneta ātruma apstākļos: parasts 2G (250 Kbps lejupielāde, 50 Kbps augupielāde, 300 ms latentums), labs 2G (450 Kbps, 150 Kbps, 150 ms), parasts 3G (750 Kbps, 250 Kbps, 100 ms) un ātrs 3G (1536 Kbps, 750 Kbps, 40 ms). Atkārtota sistēmas funkcionalitātes testēšana – ortofoto izveide, izveidoto ortofoto pārvaldība, ortofoto un sensoru datu integrācija kartē, lietotāja kartes pārskats, sensoru datu diagrammu skatīšana un koordinātu uzstādīšana – parādīja, ka sistēma pilnvērtīgi darbojas, ja lietotājam ir vismaz parasts 3G (750 Kbps, 250 Kbps, 100 ms) veikspējas interneta savienojums.

Pašreizējā sistēmas konfigurācijā WebODM spēj apstrādāt vienu ortofoto uzdevumu vienlaikus. Sistēmas prototipa ietvaros nav iekļauta vairāku paralēlu ortofoto izstrādi, tādejādi sistēmā tiek ieviests ierobežojums, lai ietaupītu WebODM apstrādes servera resursus. Gadījumā, ja prototips tiek attīstīts produkcijas stadijā, tīmekļa vietnē, kas ir atvērta publiskai lietotāju bāzei, ir iespējams pievienot vairākus WebODM apstrādes mezglus, kuri spēj veikt paralēlu ortofoto apstrādi. Pašlaik WebODM API tiek mitināts uz Ubuntu servera kam piemīt 3 VCPU (virtuāla centrālā procesora vienības) ar AMD EPYC™ 7002 sērijas procesoriem, 4 GB RAM, 80 GB NVME SSD un statiska IPV4 adrese.

## SECINĀJUMI UN PRIEKŠLIKUMI

Veidojot bakalaura darbu, darba autors ir nonācis pie sekojošiem **secinājumiem**:

- Tika veiksmīgi izveidota tīmekļa vietne, kas nodrošina lauksaimniekiem viegli lietojamu, intuitīvu ortofoto izveides procesu, sensora datu diagrammas pārskatu un augu vizuālo analīzi caur ortofoto, VARI un sensoru datu integrāciju GIS vidē.
- Galvenās sistēmas funkcionālās prasības ir: sašūta ortofoto izveide no bezpilota gaisa kuģu attēliem, ortofoto vizualizācija kartē, sensoru datu integrācija un vizualizācija kartē, sensoru datu pārskats attiecīgi izvēlētajam laikam un datumam, ortofoto kvalitātes konfigurācijas iespēja un sensora datu diagrammas pārskats.
- Veiktais eksistējošo risinājumu salīdzinājums norāda, ka eksistē daudzi risinājumi, kas daļēji atbalsta darba uzdevuma nepieciešamai funkcionalitātei, bet neviens risinājums nespēj piedāvāt pilnu funkcionalitāti vienā apvienotā, lauksaimniekiem viegli lietojamā risinājumā.
- Sensora datu API galapunktu lietotājs nevar nomainīt, jo prototipa tvērumā pašlaik sistēma nav pielāgota citu sensoru datu formātam.
- Izvēlētais tīmekļa vietnes ietvars – Streamlit ir lieliski piemērots prototipa izstrādei, jo tas nodrošina ātru tīmekļa vietnes izveidi caur iebūvētām elementu funkcijām, kuras nodrošina ievada un izvada datu funkcionalitāti.
- Tieka implementēta HTML elementu CSS formatēšana, kura iztulko Streamlit ietvara elementu tekstu latviešu valodā, jo Streamlit ietvara elementi satur tekstu, ko programmatiski nav iespējams nomainīt.
- Prototipa sistēma nav publiska un lietotājam ir nepieciešams autentificēties ar Google kontu, kuram ir pilnvarota e-pasta adrese datubāzē, jo prototipa tvērumā ir nepieciešams ietaupīt servera resursus un nodrošināt stabilu sistēmas darbību.
- Folium kartēšanas pakotne veic kritisku funkcionalitāti ortofoto vizualizācijai un sensoru ierīču koordinātas uzstādīšanā, jo pakotne nodrošina GIS vidi un interaktīvu kartes saskarni, kas spēj iegūt lietotāja izvēlētas koordinātas.
- Tīmekļa vietnes ilgtermiņa datu uzglabāšanas realizācijai tiek izmantota PostgreSQL datubāze, jo, kā norāda apskatītais pētījums, tā nodrošina ātru veikspēju un zemu latentumu.
- WebODM ir kritiska prototipa sistēmas komponente, jo WebODM nodrošina ortofoto un VARI izveidi, API autorizācijas sistēmu, lietotāju ortofoto grupējumu izveidi un projekta pārvaldes sistēmu, ortofoto izveides procesa novērošanu un ilgtermiņa ortofoto uzglabāšanu.

- Prototips sekmīgi veic sensora ierīču koordinātu operācijas, jo lietotājs spēj saglabāt, mainīt un dzēst sensoru ierīču izvēlētās koordinātas datubāzē, kā arī lokāli Streamlit ietvara sesijas atmiņā.
- Prototipa izstādes rezultātā tika izveidota viegli lietojama, intuitīva ortofoto izveides funkcionalitāte, kas spēj izveidot sašūtu ortofoto no individuāliem bezpilota gaisa kuģu attēliem un ir piemērota lauksaimnieku tehniskajām zināšanām.
- Tīmekļa vietne nodrošina skaidru ortofoto pārvaldes sistēmu, kas attēlo viskritiskāko ortofoto informāciju un nodrošina ortofoto apskates, lejupielādes un dzēšanas operācijas, bet nespēj mainīt ortofoto nosaukumu.
- Ortofoto dati – WebODM apstrādātie dati (ieskaitot VARI) un izvēlēto GeoTIFF faila dati, tiek veiksmīgi integrēti ģeogrāfiskā informācijas sistēmā, jo lietotājs spēj pārskatīt, mainīt ortofoto skata tālummaiņu un detalizēti izpētīt sašūtu lauka ortofoto kartes skatā.
- Temperatūras un mitruma sensora dati izvēlētajā datumā tiek veiksmīgi integrēti kā konfigurējami informācijas slāni GIS vidē ar iespēju vizualizēt sensoru temperatūras un mitruma mērījumus konkrētā laikā pēc vajadzības.
- Sensoru dati tiek veiksmīgi vizualizēti diagrammas skatā, jo lietotājs spēj izvēlēties sensora datu kategoriju un pārskatīt datu mērījumus laika sērijā.
- Testēšana parāda, ka sistēma ir funkcionāli pilnīga visās modernās operētājsistēmās un vairākos interneta palūkos, atbilstoša mērķiem un lietojama bez padziļinātām IT zināšanām, taču sistēmas dizains nav pilnībā pielāgots mazāku displeju ierīcēm.
- Prototips efektīvi risina dažāda mēroga lauksaimnieku problēmas lauka analīzē un lēmumu pieņemšanā, jo sistēma dod iespēju pārskatīt pietuvinātus lauka segmentus un analizēt veģetācijas veselību caur augu zaļuma indeksa ortofoto slāni kopā ar mitruma un temperatūras mērījumu informācijas slāniem, kas dod lauka attīstības pilno ieskatu.

Balstoties uz izstrādātajiem secinājumiem un darbā veiktajiem pētījumiem, darba autors izvirza **priekšlikumus**:

- Prototipa funkcionalitāte nav galīga un to var uzlabot gan vizuāli, gan funkcionāli, piemēram, pievienojot plašāku ortofoto pārvaldes sistēmas pielāgojamību.
- Integrēt iespēju izvēlēties dažādus sensora datu avotus, ne tikai vienu fiksētu API, lai uzlabotu lietotnes pielāgojamību citiem lietotājiem.
- Nodrošināt mobilajām ierīcēm draudzīgāku saskarni, lai lauksaimnieki varētu izmantot tīmekļa vietni ērtāk tieši no lauka.

- Paplašināt prototipa funkcionalitāti ar sistēmas automātisku brīdinājuma sistēmu, kas uzrauga mirkli, kad temperatūras vai mitrumu mērījumu vērtības šķērso kritiski zemu vai augstu slieksni, tādejādi nodrošinot dinamiskāku un ērtāku lauka analīzi.
- Atjaunināt autentifikācijas sistēmu, lai lietotājs spētu veikt sistēmas reģistrāciju, kuru tīmekļa vietnes administrators spēj noraidīt vai akceptēt, pat ja Google konta e-pasta adrese nav pilnvarota datubāzē, tādejādi atvieglojot lietotāja pierakstīšanās plūsmu.
- Pārrakstīt prototipu sistēmas ietvarā, kas nodrošina augstu stila pielāgojamību un paaugstinātu funkcionalitātes mērogjamību, lai nodrošinātu nākotnes sistēmas mērogjamību.

## IZMANTOTĀS LITERATŪRA UN AVOTU SARAKSTS

- [1] Snowflake Inc. Streamlit. [Tiešsaistē]. Pieejams: <https://github.com/streamlit/streamlit> [Skatīts Apr. 1, 2025]
- [2] The PostgreSQL Global Development Group. PostgreSQL: The World's Most Advanced Open Source Relational Database. [Tiešsaistē]. Pieejams: <https://www.postgresql.org/> [Skatīts Apr. 1, 2025]
- [3] Pix4D. PIX4Dfields: Drone software for agriculture mapping. [Tiešsaistē]. Pieejams: <https://www.pix4d.com/product/pix4dfields/> [Skatīts Apr. 1, 2025]
- [4] Agremo. Crop Monitoring. [Tiešsaistē]. Pieejams: <https://www.agremo.com/products/crop-monitoring/> [Skatīts Apr. 1, 2025]
- [5] QGIS Projekts. *QGIS Dokumentācija*. [Tiešsaistē]. Pieejams: <https://docs.qgis.org/3.40/en/docs/index.html> [Skatīts Apr. 1, 2025]
- [6] Restack. (2024, Nov.). *Streamlit vs Flask vs Django comparison*. [Tiešsaistē]. Pieejams: <https://www.restack.io/docs/streamlit-knowledge-streamlit-vs-flask-vs-django> [Skatīts Apr. 28, 2025]
- [7] R. Story. *Folium*. [Tiešsaistē]. Pieejams: <https://python-visualization.github.io/folium/latest/> [Skatīts Apr. 1, 2025]
- [8] A. Ouda un S. V. Sanket. (2024, Okt.). "A Performance Benchmark for the PostgreSQL and MySQL Databases," *Future Internet* [Tiešsaistē]. vol. 16, issue 10. Pieejams: <https://doi.org/10.3390/fi16100382> [Skatīts Apr. 1, 2025]
- [9] OpenDroneMap Komanda. *NodeODM*. [Tiešsaistē]. Pieejams: <https://opendronemap.org/nodeodm/> [Skatīts Apr. 1, 2025]
- [10] OpenDroneMap Komanda. *WebODM GitHub*. [Tiešsaistē]. Pieejams: <https://github.com/OpenDroneMap/WebODM> [Skatīts Apr. 1, 2025]
- [11] S. Knight. (2023, Feb. 15). *Most Popular Email Providers by Number of Users (2023)*. [Tiešsaistē]. Pieejams: <https://www.sellcell.com/blog/most-popular-email-provider-by-number-of-users/> [Skatīts Maijs. 11, 2025]
- [12] C. S. Kummarapurugu. "A Comparative Analysis of OAuth 2.0 and OpenID Connect for Identity Federation in Cloud Environments," *World Journal of Advanced Research and Reviews*, vol. 1, pp. 54-60, Marts, 2019.
- [13] OpenDroneMap Komanda. *WebODM Documentation*. [Tiešsaistē]. Pieejams: <https://docs.webodm.org/> [Skatīts Maijs. 10, 2025]

[14] OpenDroneMap. *sfm-algorithm.* [Tiešsaistē]. Pieejams: <https://docs.opendronemap.org/arguments/sfm-algorithm/> [Skatīts Maijs. 20, 2025]

# PIELIKUMI

Pielikums Nr. 1

```
def pieprasit_odm(metode, url, dati=None, faili=None, stream=False,
meginajumi=0):
    meginajumi += 1
    if meginajumi == 5: # Izseko vai pieprasījums ir atkārtojies 5 reizes
        st.error("Neizdevās atkārtoti iestatīt WebODM JWT žetonu.")
    # Tieka iegūta autorizācijas galvene
    galvene = json.loads(st.session_state.sikdatne["galvene"])
    try:
        if metode == "GET":
            atb = requests.get(url, headers=galvene, stream=stream)
        elif metode == "POST":
            atb = requests.post(url, headers=galvene, data=dati, files=faili)
        else:
            raise ValueError("Neatbalstīta metode")

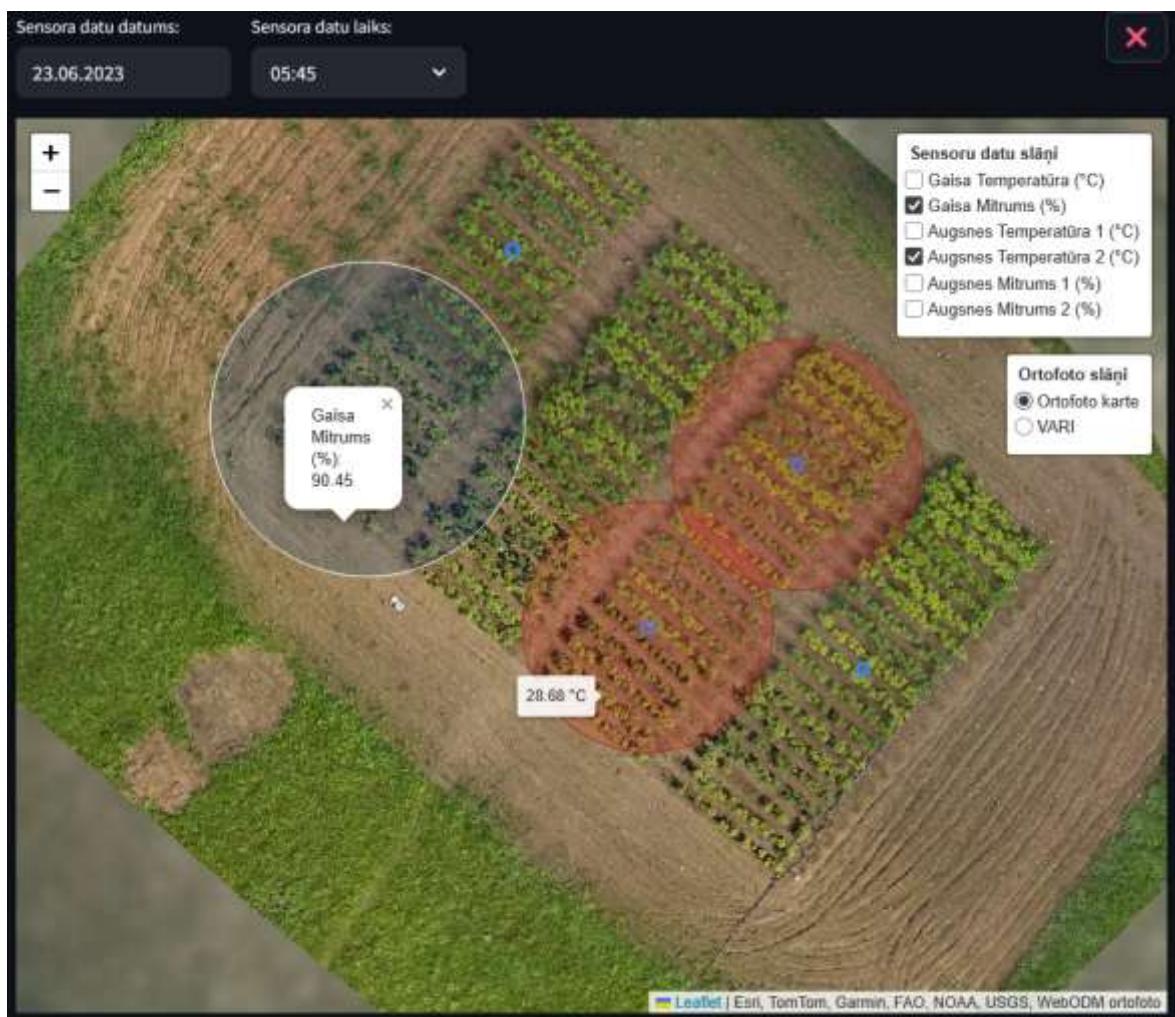
        atb.raise_for_status()
        return atb
    except HTTPError as e:
        if e.response.status_code == 403: # Ja autorizācijas žetona derīguma
            rermīns ir beidzies
            ir_galvene_iestatita = iestatit_galveni()

        if ir_galvene_iestatita:
            # Tieka atkārtoti veikts WebODM pieprasījums
            if metode == "GET":
                return pieprasit_odm("GET", url, stream=stream,
meginajumi=meginajumi)
            elif metode == "POST":
                return pieprasit_odm("POST", url, dati=dati, faili=faili,
meginajumi=meginajumi)
            else:
                st.error("Atkārtotā WebODM pieprasījumā neizdevās iestatīt JWT
žetonu.")
        else:
            st.error(f"Kļūda pieprasījumā: {e}")
```

*Pielikums Nr. 2*

Bakalaura darbā tika apskatīta tikai daļa no kopējā koda. Darba izstrādes gaitā darba autors izmantoja GitHub servisu, lai uzglabātu pilno prototipa kodu.

Pilnais kods ir pieejams GitHub repozitorijā: <https://github.com/jacob-konosx/BakalauraDarbs>





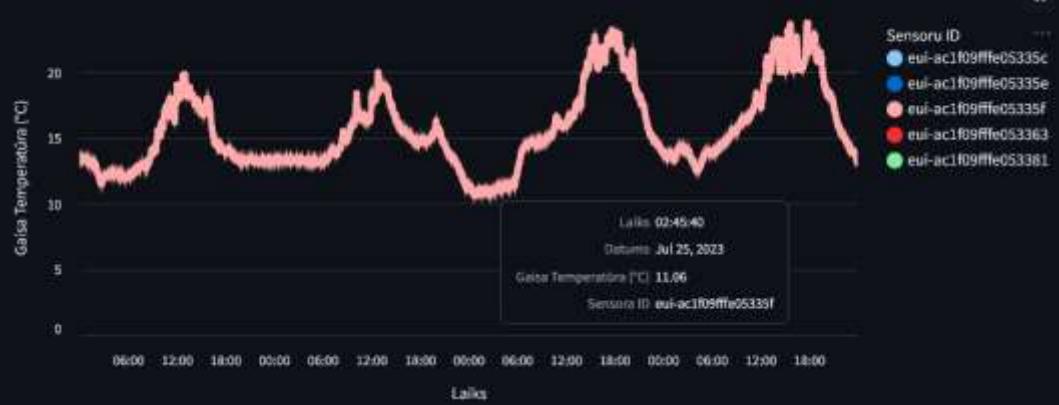
## Sensora datu diagramma

Ievadiet sensoru datuma diapzonu:

23.07.2023 – 27.07.2023

Datu kategorijas:

Gaisa Temperatūra (°C)



## **GALVOJUMS**

Ar šo es, Jēkabs Konošonoks, galvoju, ka šis bakalaura darbs ir manis paša patstāvīgi izpildīts oriģināls darbs. Visi informācijas avoti, kā arī no tiem ņemtie dati un definējumi ir norādīti darbā. Šis darbs tādā vai citādā veidā nav iesniegts nevienai citai pārbaudījumu komisijai un nav nekur publicēts.

Esmu informēts (-a), ka mans bakalaura darbs tiks ievietots un apstrādāts Vienotajā datorizētajā plašiā kontroles sistēmā plašiā kontroles nolūkos.

202\_\_\_. gada \_\_\_\_\_

(paraksts)

Es, Jēkabs Konošonoks, atļauju Ventspils Augstskolai savu bakalaura darbu bez atlīdzības ievietot un uzglabāt Latvijas Nacionālās bibliotēkas pārvaldītā datortīklā Academia ([www.academia.lndb.lv](http://www.academia.lndb.lv)), kurā tie ir pieejami gan bibliotēkas lietotājiem, gan globālajā tīmeklī tādā veidā, ka ikviens tiem var piekļūt individuāli izraudzītā laikā, individuāli izraudzītā vietā.

Piekriņu \_\_\_\_\_

Nepiekriņu \_\_\_\_\_

202\_\_\_. gada \_\_\_\_\_