# AI-Driven Forest Management: Leveraging Remote Sensing and Machine Learning for Sustainable Forestry

Adviser: Prof. Michael Lewis

Aliza Momysheva
*School of Engineering and Digital Sciences*
*Nazarbayev University*
Astana, Kazakhstan
aliza.momysheva@nu.edu.kz

Ariana Sadyr
*School of Engineering and Digital Sciences*
*Nazarbayev University*
Astana, Kazakhstan
ariana.sadyr@nu.edu.kz

Dulat Rakhymkul
*School of Engineering and Digital Sciences*
*Nazarbayev University*
Astana, Kazakhstan
dulat.rakhymkul@nu.edu.kz

***Index Terms*—Artificial Intelligence, Forestry, Vegetative Indices, Satellite Imagery**

## I. Executive Summary

Nowadays, we rely on advanced technologies for sustainable forest management tasks to address the challenges of monitoring vast and ecologically diverse forests. There are no existing scalable and automated solutions for assessing forest conditions in Kazakhstan, making monitoring and preserving forest environments difficult. This project's main goal is to develop a comprehensive, computing-based application that integrates aspects of remote sensing, geospatial data processing, and artificial intelligence to support modern tools for forest monitoring. The core objective is to construct a dataset for specific forest regions in Kazakhstan using biweekly satellite images from Sentinel-2 and LANDSAT satellites. Specifically, the dataset consists of forest masks generated through the threshold classification of vegetative indices, such as NDVI, and a range of vegetation indices for assessment of forest health and disturbance detection. Data gaps caused by cloud cover were addressed using temporal interpolation and reprojection techniques to produce complete forest masks. Moreover, the application includes a chatbot based on a retrieval augmented generation (RAG) system, which enables users to query the system by passing their questions as prompts and receiving contextualized responses from the database. The chatbot's role is to assist with forest management questions for the user, leveraging modern smart query systems in combination with Large Language models (LLM). The resulting mobile application provides functionalities such as forest mask visualization, deforestation and fire detection, and access to vegetation metrics and their analysis. The application is designed to be intuitive and user-friendly, ensuring ease of use for all stakeholders, regardless of their technical background. To assess the quality of the application, the satisfaction levels of users are evaluated through their direct feedback. This work illustrates a comprehensive approach for designing, implementing, and validating a forest management application, with scalable potential for broader use in making decisions for environmental challenges.

## II. Introduction

Forest management is essential worldwide for monitoring forest health, carbon deposits, the shift of climate zones, and other aspects that can be important for the support of ecological and economic sustainability. In Kazakhstan, where forested regions are mostly large and remote territories, there is a need for efficient and data-driven solutions for forest monitoring. Traditional on-site forest management in Kazakhstan is still developing, so there is insufficient data regarding local forests. This project is motivated by the urgent need for an automated, scalable solution that delivers precise, real-time insights about forests. The proposed solution utilizes Sentinel-2 satellite imagery to calculate NDVI (Normalized Difference Vegetation Index) values, which are a vegetative index that measures the density and health of vegetation, with a threshold of 0.2–0.3 for forest detection, supplementing missing data with LANDSAT satellite images harmonized through a nearest-neighbor reprojection method. The binary forest masks are stored as numpy arrays in a database for each biweekly interval. Due to noise and false positives in the winter months (November - February), the analysis is limited to data from March to October. The system identifies deforestation events in real-time by comparing successive forest masks, providing immediate insights into landscape changes. The project also monitors fire events, tracking the progression of burned areas in Semey Ormany using Normalized Burn Ratio (NBR) metrics, which is one of the vegetative indices used in the project. This report discusses past works and then provides a detailed description of our solution. Then, it provides details of the project execution, evaluation methods, a conclusion of the work done, and a discussion regarding future work.

## III. Background and Related Work

Our project relies on satellite imagery for forest management tasks as it offers a non-invasive and more extensive approach for highly precise continuous monitoring. It provides insights into forest health by delivering time series data and analysis of vegetative indices over specific forest areas.

### *A. Relevant projects*

During our initial research, we identified several projects and studies that address various aspects of forest monitoring, carbon accounting, and other forestry practices. Most are forest and agriculture-related projects, and they sometimes publish their methodologies on their official pages. They often utilize satellite imagery for forest segmentation, above-ground biomass calculation, and deforestation detection.

One of the most notable projects was Ctrees because it made various literature related to their project methodologies publicly available on their website. For instance, they leveraged Landsat and NICFI satellite imagery with a U-Net AI model to enhance forest segmentation and deforestation monitoring. Also, they utilized data fusion models that integrate data from various sources, including LIDAR, optical, and microwave imagery, and in situ inventory plots, to provide accurate estimates of carbon [1]–[3].

The Pachama project integrates similar satellite data, field plots, and 3D airborne lidar imaging to map forest carbon. Pachama's methodology uses LANDSAT, PALSAR, and GEDI satellite data to capture forest structure and ”greenness.”

Another project called Climate Action Data Trust provides a centralized information system on carbon credit projects, offering a standardized procedure for calculating and verifying emission reductions. However, it does not publicly expose information from private and national registers. It does not mention the usage of any AI architectures, which we focused on during the research.

We noticed that platforms like Sylvera and BeZero use machine learning to assess carbon projects. Sylvera employs terrestrial and airborne laser scanners and multi-scale LiDAR technology to estimate biomass and carbon changes. BeZero uses data from spaceborne LiDAR, synthetic-aperture radar, and multispectral imagery to rate carbon projects.

### *B. Satellite imagery*

Several papers have been found that dive more into the methodologies of satellite image processing. For instance, some research also leveraged LANDSAT, MODIS, and Sentinel-2 satellites to extract images of forests and used them to train AI models such as U-Net and Random Forest for forest segmentation into categories like primary and secondary forests, roads, and burns. Also, they employed vegetative indices from those images to assess forest health [5]–[10]. Other research also focused on machine learning-assisted remote forestry health monitoring, and in their papers, they provided insights into modern approaches to forest management [11], [12].

However, none of these sources mentioned forests in Kazakhstan, and we could not find any relevant datasets. Thus, we decided to research which satellite imagery we can utilize for our needs. We composed a table with brief descriptions of all satellite imagery sources we found in Table I. According to the table and additional information regarding the ease of use and data availability, we decided to use Sentinel-2 as our primary source of satellite images and LANDSAT for supplementary data. The main reasons are that both satellite images are open source, and it is easy to access their database. Sentinel-2 provides high-resolution images up to 10m per pixel. Both satellites cover Kazakhstan forests and have continuous records of them over long periods. Other satellites may provide even more and better information, but most of their databases are private or fee-based, and they do not cover our requirements. For instance, one of the private satellite imageries we would like to use is GEDI. It is known for its use of LiDAR scanner, which can help validate specific forest health and compare it to vegetative indices derived from satellite images. However, as it does not have an open-source API service, we used other high-quality satellite imageries as alternatives.

### *C. Comparison to other projects*

Our project's methodology is based on leveraging Sentinel-2 and LANDSAT satellite imagery to create biweekly time series data, which allows detailed monitoring of forests by tracking changes in their coverages and analyzing vegetative indices to get insights into their health. This approach is intended to capture deforestation, fire events (e.g., of Semey Ormany forest), and vegetation dynamics with high spatial and temporal resolution of images. Spectral bands of images extract relevant environmental insights, which have been well-documented in various scientific papers. To use this approach, we first gathered all available data from satellites for specific forest areas in Kazakhstan and created our dataset. In addition, we integrated an AI Chatbot with RAG functionality to serve as a smart query engine, which provides a user-friendly interface for answering users' questions. The main role of the Chatbot is to provide more information for various stakeholders and assist them in better understanding our product.

In contrast, projects like Ctrees demonstrate the power of combining ML models and satellite imagery for forest analysis. They use data fusion models to predict carbon stocks but may face limitations in data availability for specific regions. For instance, they used the NICFI satellite, which covers very limited regions, such as some cities in Brazil [1]–[3]. Others, like Climate Action Data Trust, also offer carbon

TABLE I: Comparison of Satellite Imagery Sources

| **Satellite** | **Description** | **Key Features** | **Applications** |
|---|---|---|---|
| Planet Labs | Provides high-resolution imagery through PlanetScope, RapidEye, and SkySat satellites. | • Daily global coverage<br>• Varying spatial resolutions | Agriculture, forestry, and land use monitoring |
| Landsat (NASA) | Long-standing satellite program offering global coverage at 30m spatial resolution. | • Over 40 years of historical data<br>• Monthly global coverage | Land use, vegetation monitoring, and environmental change analysis |
| Sentinel-2 (ESA) | Part of the Copernicus program, providing optical imagery with 13 spectral bands. | • Spatial resolutions: 10m, 20m, 60m<br>• Swath width: 290 km | Vegetation monitoring, soil and water cover assessment, land classification |
| MODIS (Terra & Aqua) | Captures continuous fields of vegetation data. | • Spatial resolution: 250m<br>• Large-scale monitoring | Environmental monitoring, vegetation dynamics, and climate studies |
| GEDI (NASA) | LiDAR system producing precise 3D forest structure models. | • High accuracy in biomass and carbon storage assessment<br>• Habitat quality analysis | Biomass assessment, carbon storage, and forest structure analysis |
| PALSAR (ALOS, Japan) | Radar-based sensor for analyzing surface structures and vegetation. | • L-band Synthetic Aperture Radar<br>• Polarimetric data for feature analysis | Surface structure analysis, vegetation monitoring, and forestry |

accounting frameworks but do not provide analytics as it does not use any AI technologies. Pachama also provides some carbon mapping but does not have detailed information on other aspects, such as deforestation or fire detection. Platforms like Sylvera and BeZero excel in carbon project assessment but are not optimized for real-time forest monitoring.

A key differentiator of our project is the focus on developing a region-specific dataset for Kazakhstan's forests. As noted in our report, publicly available labeled datasets suitable for our specific scenario do not exist. Therefore, we created our own dataset using satellite images and manual labeling using custom Python scripts tailored to Kazakhstan's forests' unique characteristics. Additionally, another main feature of our project is that we provide a real-time AI assistant with an implemented smart query system, which makes our product more user-friendly for any kind of stakeholder, such as users without domain-specific knowledge of forest management.

## IV. Project Approach

### A. Study Area

We selected forests from North Kazakhstan and East Kazakhstan due to their extensive forest cover and ecological diversity, which make these regions particularly significant for the project. Our study focused on four distinct forest areas across these regions, establishing a study timeline from 2020 to 2025 to capture seasonal and annual variations in forest dynamics. Given the lack of existing appropriately labeled data for Kazakhstan forests, we gathered satellite images and processed them for our dataset.

We created a dedicated JSON file for each forest to organize and manage our spatial and temporal data systematically. These JSON files include a unique identifier (id) for each forest, a bounding box (bbox) that delineates the precise geographical extent of the study area, and two sets of selected dates: one for Sentinel-2 imagery (s2_dates) and another for LANDSAT imagery (landsat_dates). The selected dates correspond to periods when high-quality, cloud-free images are available, ensuring our mapping and analysis consistency. This structured approach facilitates a robust comparison between different satellite sources and streamlines data processing and enhances the reproducibility of our forest monitoring and deforestation analyses over the defined study period.

### B. Data Sources

In this project, Sentinel-2 imagery from the COPERNICUS/S2_SR collection is our primary data source, offering high-resolution (10 m) atmospherically corrected surface reflectance data ideal for detailed forest monitoring. Building on previous work, we selected two images per month—one from early in the month and one from mid-month—to effectively capture temporal changes in forest conditions. Although Sentinel-2 is designed to revisit the same location approximately every 5 days, factors such as cloud cover and adverse seasonal conditions can render some images unusable, leading to gaps in the data. To address these gaps, we incorporate LANDSAT imagery from the LANDSAT/LC08/C02/T1_L2 collection, which provides atmospherically corrected surface reflectance and surface temperature data at a 30 m resolution. While LANDSAT offers a coarser spatial resolution than Sentinel-2, its inclusion is critical for maintaining consistent temporal coverage when high-quality Sentinel-2 images are unavailable. By integrating these two datasets, we balance the advantages of Sentinel-2's fine spatial detail with LANDSAT's reliable availability, ensuring a robust and continuous record of forest dynamics over our study period.

### C. Data Preprocessing

In the data pre-processing part, we mostly relied on vegetative indices, which are important for forest monitoring tasks via satellite imagery. Vegetative indices are derived from satellite imagery, and they enable consistent large-scale monitoring of vegetation health, water stress, biomass, and fire impact. They help a lot to detect changes in forest conditions over time and support stakeholders in making certain decisions for forest preservation. In our project, we researched and implemented a wide range of indices using Sentinel-2 and LANDSAT spectral bands. Table II summarizes the key indices used, along with their primary focus.

These indices provide details for a comprehensive analysis of diverse forest conditions, so they help with deforestation detection, moisture tracking, and post-fire recovery assessment. They are the key elements for forest management tasks leveraging satellite imagery.

*1) **Cloud Filtering and Masking**:* In the data preprocessing stage, we first apply a cloud threshold of 20% to both Sentinel-2 and LANDSAT imagery to filter out scenes with excessive cloud cover, ensuring that only images with acceptable clarity are considered. This filtering lists dates when the imagery meets our basic quality criteria. However, to further ensure the reliability of our analysis, we manually review the images corresponding to these dates and select the clearest ones. This two-step approach is the initial automated filtering followed by manual verification and it helps us to compile a high-quality dataset that accurately represents the forest conditions in our study regions.

In our data preprocessing workflow, rather than removing pixels identified as clouds or water, we assigned them a special classification to prevent these areas from being mistakenly identified as deforestation in our temporal analysis. For Sentinel-2 imagery, we utilized the Scene Classification Layer (SCL) to identify pixels corresponding to clouds, cloud shadows, and water, labeling them with a unique class. Similarly, we applied the quality assessment bands for LANDSAT data to mark cloud- and water-affected pixels with a distinct classification. This approach preserves complete spatial information while clearly distinguishing atmospheric or water-related artifacts, ensuring that such areas are not erroneously interpreted as deforestation when comparing sequential forest masks.

*2) **Forest Masking**:* In our forest masking process, we employed the Normalized Difference Vegetation Index (NDVI) as a key indicator of vegetation health and density. NDVI is calculated using the formula (NIR - Red) / (NIR + Red), where NIR represents the near-infrared reflectance and Red represents the red band reflectance. The index yields values ranging from -1 to 1, with higher values typically indicating healthier, denser vegetation. After some observations and relying on the previous research, we chose a threshold range of 0.2 to 0.3 for forest classification. This threshold was selected because forested areas generally exhibit NDVI values above this range, allowing us to effectively differentiate true forest cover from non-forest areas, which often register lower NDVI values. Using this threshold, we aim to minimize misclassification, ensuring that areas with sparse vegetation or other land cover types are not erroneously identified as forests. Our final forest mask, such as in Figure 1, has three classes, where 1 indicates forest, 2 represents cloud or water, and 0 represents non-forest.

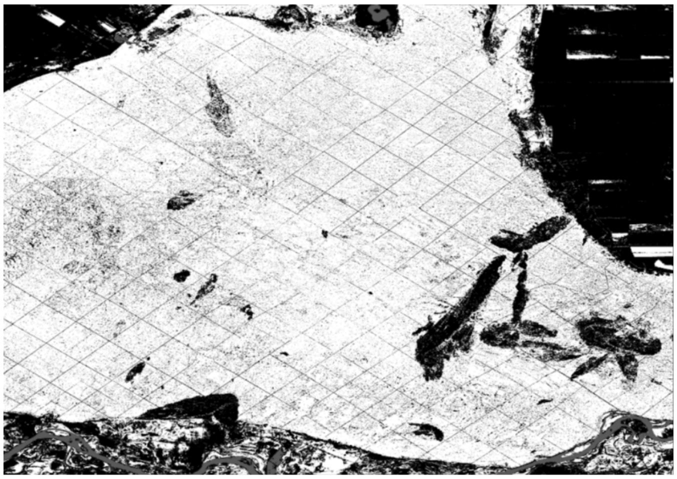


Fig. 1: Forest mask for Semey Ormany

*3) **Burned Area Masking**:* In our burned-area masking process, we employed the Normalized Burn Ratio (NBR) as a key indicator of fire impact. NBR is calculated using the formula

$$\mathrm{NBR} = \frac{\mathrm{NIR} - \mathrm{SWIR}_2}{\mathrm{NIR} + \mathrm{SWIR}_2} \tag{1}$$

where NIR represents the near-infrared reflectance and $\mathrm{SWIR}_2$ the short-wave infrared reflectance. To capture the Semey Ormany fire event, we selected a three-day temporal window acquiring pre- and post-fire images exactly three days apart and computed the change in NBR as:

$$\Delta\mathrm{NBR} = \mathrm{NBR}_{\mathrm{post}} - \mathrm{NBR}_{\mathrm{pre}} \tag{2}$$

which yields values from –2 to +2, with more negative values indicating greater burn severity. Based on empirical observation and relevant literature, we chose a $\Delta$NBR threshold of –0.10, since burned areas generally exhibit $\Delta$NBR values below this cutoff. Applying this threshold allowed us to effectively distinguish burned surfaces from unburned land, minimizing misclassification of vegetation regrowth or moisture-induced fluctuations. Our final burned mask as Figure 2 comprises two classes: 1 for burned area and 0 for unburned ground.

*4) **Gap Filling**:* When mapping forest areas, occasional cloud cover can obscure parts of the imagery, complicating the accurate classification of forested regions. To address this, we implemented a temporal interpolation strategy for cloudy pixels. Specifically, we identify the nearest clear images before and after the date in question for any pixel classified as cloudy. If the corresponding pixels in these clear images meet our forest criteria (by exceeding the NDVI threshold), we infer that the forest is likely present on the cloudy date and update

TABLE II: List of Vegetation Indices Descriptions

| Index | Full Name | Primary Use | Formula |
|---|---|---|---|
| NDVI | Normalized Difference Vegetation Index | General vegetation health and greenness | (NIR – R) / (NIR + R) |
| EVI | Enhanced Vegetation Index | Improved sensitivity in high biomass areas | 2.5 × (NIR – R) / (NIR + $C_1$ × R – $C_2$ × B + L) |
| NDWI | Normalized Difference Water Index | Water content in vegetation | (NIR – SWIR) / (NIR + SWIR) |
| NBR | Normalized Burn Ratio | Burn severity and fire detection | (NIR – SWIR) / (NIR + SWIR) |
| SAVI | Soil Adjusted Vegetation Index | Vegetation in areas with exposed soil | (NIR – R) / (NIR + R + L) × (1 + L) |
| GNDVI | Green NDVI | Chlorophyll content and vegetation stress | (NIR – G) / (NIR + G) |
| SIPI | Structure Insensitive Pigment Index | Pigment concentration in vegetation | (NIR – B) / (NIR – R) |
| MGRVI | Modified Green-Red Vegetation Index | Vegetation density and structure | ($G^2$ – $R^2$) / ($G^2$ + $R^2$) |
| TGI | Triangular Greenness Index | Chlorophyll content using visible bands | G – 0.39R – 0.61B |
| VARI | Visible Atmospherically Resistant Index | Vegetation greenness using visible bands | (G – R) / (G + R – B) |
| GRVI | Green-Red Vegetation Index | General vegetation vigor | (G – R) / (G + R) |
| SR | Simple Ratio | Vegetation density and productivity | NIR / R |
| CI | Chlorophyll Index | Estimating chlorophyll concentration | (NIR / G) – 1 |
| MSR | Modified Simple Ratio | Enhancing SR for biomass estimation | [(NIR / R) – 1] / [sqrt(NIR / R) + 1] |
| OSAVI | Optimized SAVI | Better correction for soil brightness | (NIR – R) / (NIR + R + 0.16) |
| NDMI | Normalized Difference Moisture Index | Vegetation water content | (NIR – SWIR) / (NIR + SWIR) |
| MSAVI | Modified SAVI | Reduces soil influence in NDVI | (2 × NIR + 1 – sqrt((2 × NIR + 1)$^2$ – 8 × (NIR – R))) / 2 |
| NDRI | Normalized Difference Red Edge Index | Red edge vegetation stress detection | (R – G) / (R + G) |
| RECI | Red Edge Chlorophyll Index | Chlorophyll content using red-edge bands | (NIR – RE) / RE |

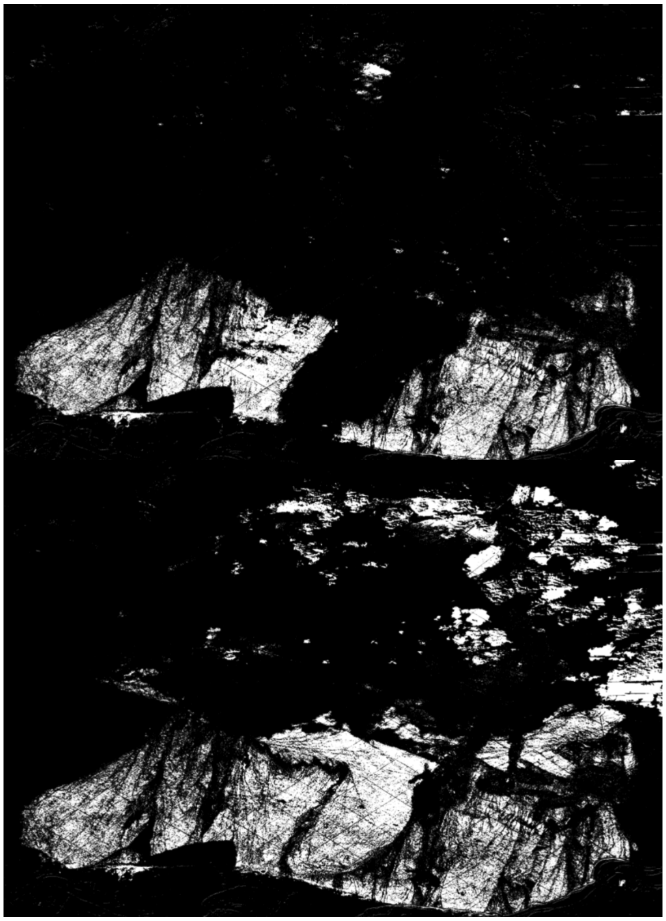


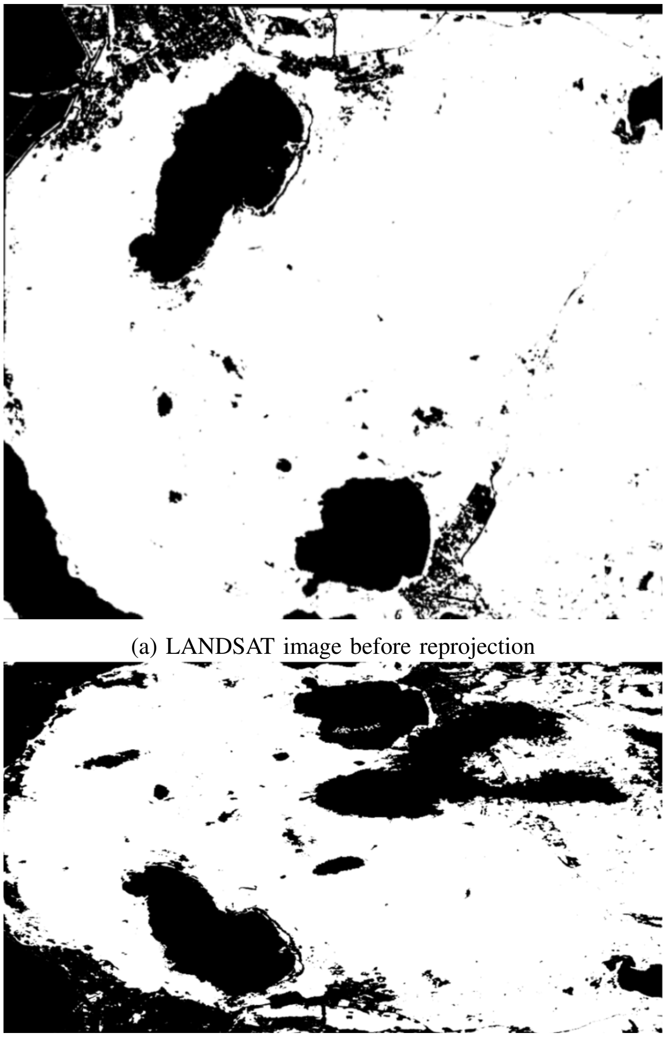
Fig. 2: Burned area mask images of Semey Ormany forest

the cloudy pixel to a forest classification (marked as 1). This method ensures that transient cloud cover does not lead to the misclassification of forested areas, thereby enhancing the continuity and reliability of our forest mapping process.

*5) **LANDSAT Reprojection**:* Due to differences in spatial resolution between Sentinel-2 and LANDSAT imagery, the reprojection of the LANDSAT data was necessary to ensure compatibility and accurate comparison. Sentinel-2 provides imagery at a 10 m resolution, while LANDSAT images are at 30 m. To harmonize these datasets, we applied a nearest neighbor reprojection method to resample LANDSAT imagery to match the finer resolution of Sentinel-2. This approach is particularly effective for our application because it preserves the categorical integrity of the data, such as binary forest masks and special classes for cloud and water, by preventing the interpolation errors that could arise from other resampling techniques. The reprojection not only aligns the spatial dimensions and coordinate systems of the two data sources but also ensures that our forest, deforestation, and fire mapping analyses remain consistent across different sensor platforms and temporal intervals. The visual example of LANDSAT images before and after reprojection can be seen in Fig 3

### D. Dataset Size

The dataset consists of Sentinel-2 and LANDSAT satellite images downloaded from Google Earth Engine in TIFF format. It allows us to use spectral band values to calculate and store different vegetative indices in the database for the work of our application. Our dataset consists of approximately 300 biweekly satellite images for chosen forest regions in



(a) LANDSAT image before reprojection

(b) LANDSAT image after reprojection

Fig. 3: Comparison of LANDSAT images before and after reprojection

Kazakhstan which are North Kazakhstan, East Kazakhstan, Semey Ormany, and Semey Ormany 2. The size of the dataset is 1.3 GB and it is stored in the Google Drive directory which we use in the application via custom Python scripts.

### *E. RAG-based Chatbot integration*

The final version of the RAG-based Chatbot in our Forest Management Application was integrated mainly using OpenAI API endpoints. There are not enough domain-specific literature and datasets on the forests of Kazakhstan to train an AI model from scratch. Therefore, we decided to use existing large language models such as GPT-4 because it is a state-of-the-art model that can be fine-tuned for specific tasks and it shows an exceptional performance.

The architecture of the RAG-based Chatbot consists of three primary parts: router, context builder, and LLM API endpoint, as seen in Figure 4. The router decides which specific data to query from our application's database and then sends it to the context builder. Context builder preprocesses information from the database and then adds it as an additional context to the user's prompt. Then, this enhanced prompt is sent to any LLM, which generates a response that is shown to the user.

The basic RAG systems query vector-based databases and use a similarity score to retrieve chunks of probably relevant data. In our case, we query the SQLite database intending to get specific information, which makes the system more efficient and reliable.

### *F. Software Architecture*

The software architecture of our system follows a monolithic structure, where the forest analysis and chatbot services are integrated into a single backend application built with Django and the Django REST Framework. The backend exposes a set of RESTful APIs that interact with the Flutter-based frontend, enabling users to request and visualize forest-related data or interact with the chatbot. The project is divided into two main services: the Forest Analysis Service, which handles the retrieval, processing, and visualization of satellite imagery and derived indices, and the Chatbot Service, which leverages retrieval augmented generation (RAG) to answer domain-specific questions. We use a lightweight SQLite database for storage, which stores metadata, forest indices, and image paths. Google Earth Engine is used to access and process large-scale satellite imagery. At the same time, Google Drive serves as a cloud storage solution for high-resolution TIFF files of masks, which are processed into PNG files saved in the database. This architecture allows for a modular yet cohesive system that is easy to maintain and extend, especially useful in a research-oriented setting. Figure 5 visually describes the software architecture of the application.

The Flutter frontend is a cross-platform interface, which allows a seamless deployment on Android and iOS devices from a single codebase. It communicates with the Django REST API to retrieve and display forest data, interactive visualizations, and chatbot responses. Flutter ensures a responsive and intuitive user experience, supporting real-time updates and user-friendly interaction with satellite imagery, forest metrics, and conversational features. With its high performance and customizable widgets, Flutter complements the backend's functionality by providing a modern, accessible UI for both field researchers and general users.

### *G. Database System*

The database design uses a normalized structure where *ForestModel* is the central table. Each forest is linked to multiple entries in the *IndicesModel* (e.g., NDVI, NBR over time), *ForestMaskModel* (classified forest cover masks), and *BurnedAreaModel* (detected burned areas). Timestamp fields across the models support chronological analysis. The chatbot system follows a similar relational design with *MessageModel* referencing the User table. Each message stores metadata such as sender role and creation time, enabling seamless chat functionality. These relational models support efficient spatial,

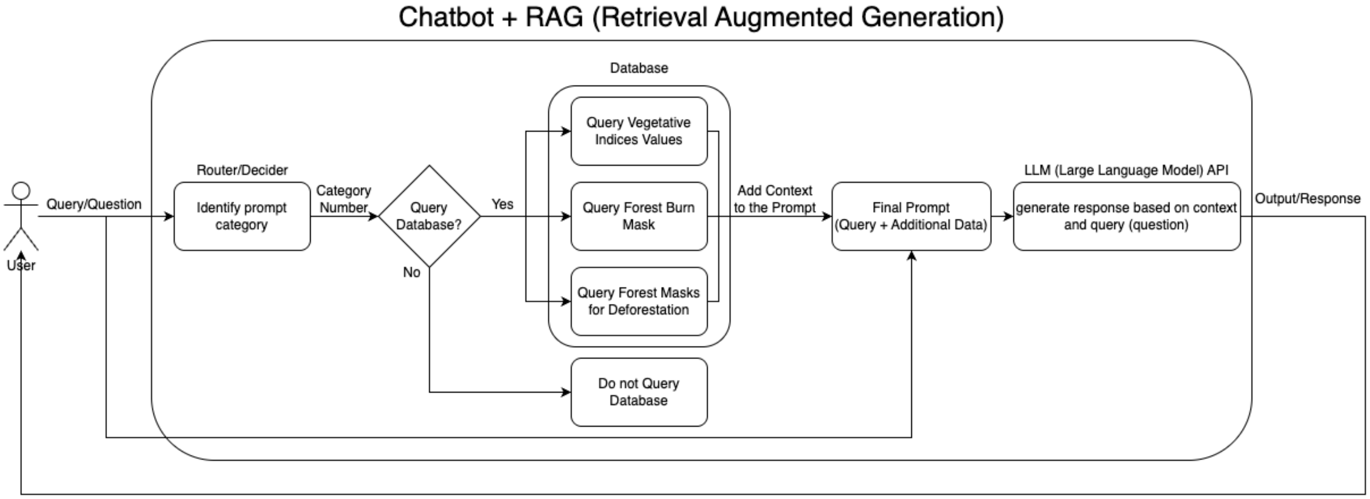

Chatbot + RAG (Retrieval Augmented Generation)
User
Query/Question
Router/Decider
Identify prompt category
Category Number
Query Database?
Yes
No
Database
Query Vegetative Indices Values
Query Forest Burn Mask
Query Forest Masks for Deforestation
Do not Query Database
Add Context to the Prompt
Final Prompt (Query + Additional Data)
LLM (Large Language Model) API
generate response based on context and query (question)
Output/Response

Fig. 4: RAG-based Chatbot architecture

temporal, and interaction-based querying across analysis and communication systems. The detailed UML schema of the database is shown in Figure 6

*ForestModel:*

- **name (str):** Name given to the forest area.
- **unique_id (str):** A manually or algorithmically assigned unique identifier, possibly used for external referencing or traceability.
- **polygon_coors (JSON):** JSON field storing geographical coordinates representing the forest polygon boundaries.

*IndicesModel:*

- **forest (FK):** Foreign key linking to ForestModel; indicates which forest the index belongs to.
- **name (str):** Name of the vegetation index (e.g., NDVI, NBR).
- **value (float):** Computed value of the index.
- **timestamp (datetime):** Date and time the index value was recorded (used for time-series analysis).

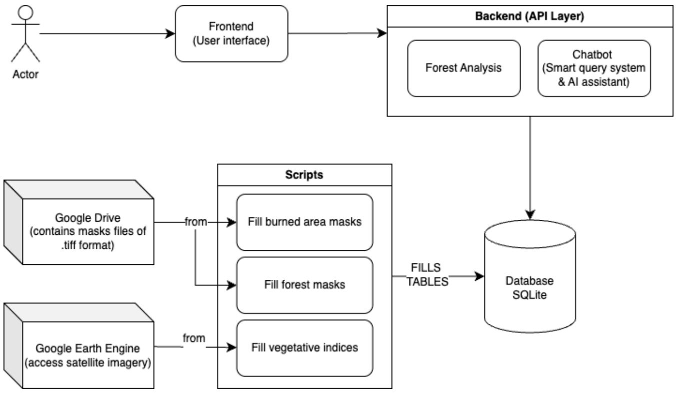

Actor
Frontend (User interface)
Backend (API Layer)
Forest Analysis
Chatbot (Smart query system & AI assistant)
Scripts
Fill burned area masks
Fill forest masks
Fill vegetative indices
Google Drive (contains masks files of .tiff format)
Google Earth Engine (access satellite imagery)
from
from
FILLS TABLES
Database SQLite

Fig. 5: Software Architecture Design

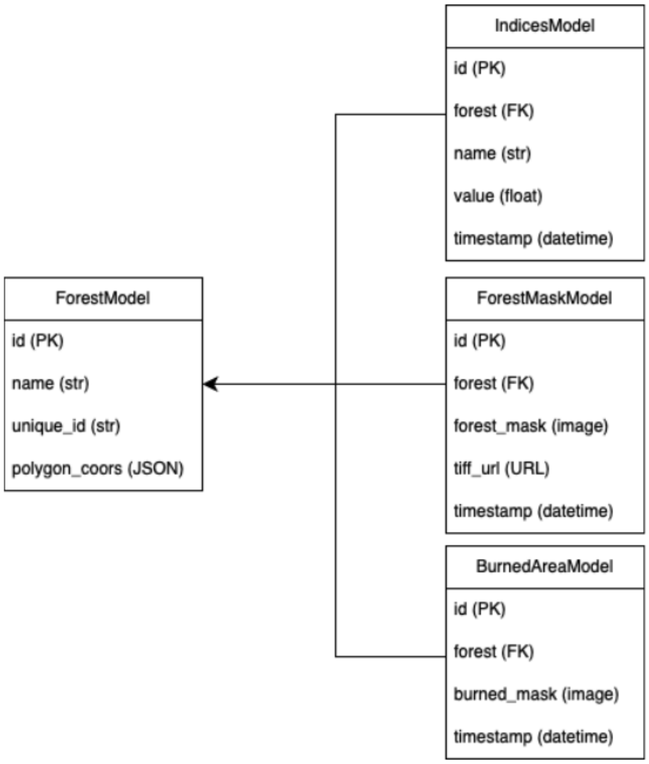


Fig. 6: UML diagram of Forest Analysis Database Schema

*ForestMaskModel:*

- **forest (FK):** Reference to the related forest.
- **forest_mask (ImageField):** Path to the classified binary image showing forest areas.
- **tiff_url (URL):** Link to the original GeoTIFF file stored on a platform like Google Drive.
- **timestamp (datetime):** Date when the mask was gen-

erated, typically matching the satellite image acquisition time.

*BurnedAreaModel:*

- **forest (FK):** Reference to the related forest.
- **burned_mask (ImageField):** Path to the burned area mask image file.
- **timestamp (datetime):** Date of fire event detection or satellite image capture.

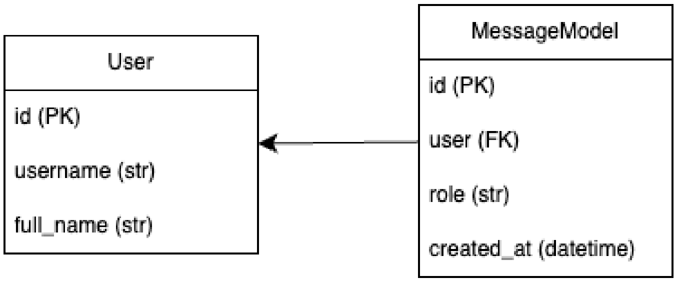


Fig. 7: UML diagram of Chatbot Message Database Schema

The diagram in Figure 7 illustrates the relationship between the User and MessageModel tables in the database. The User table in the diagram represents Django's built-in User model from django.contrib.auth.models. This model provides standard authentication and user management functionality. Each MessageModel instance is linked to a User via a foreign key, indicating that a user can send multiple messages.

*MessageModel table:*

- **user (FK):** Foreign key linking the message to the user who sent it (can be null for system messages).
- **role (str):** Indicates whether the sender is the 'user' (human user) or 'assistant' (AI chatbot). This supports role-based display logic in the chat interface.
- **text (TextField):** The actual content of the message.
- **created_at (datetime):** Timestamp marking when the message was created.

### H. Algorithms

Our system employs several remote sensing and data processing algorithms for forest monitoring and chatbot interaction.

*1) **Calculation of the mean value of vegetational indices**:* The code for computing vegetation indices uses satellite-specific spectral bands and applies mathematical formulas to calculate new image layers. For each index (e.g., NDMI, MSAVI), the relevant bands (such as NIR, RED, SWIR) are selected from the satellite image and inserted into the formula. These expressions are applied to every pixel in the image, producing a new band representing that index, which is then added to the image.

The calculate_mean_indices function computes the average (mean) value of each selected index within a specified geographical region (bbox). It first selects the relevant index bands from the image and then applies a mean reducer over that region. This operation calculates a spatial mean for each index over the given area and returns the result as a dictionary with index names and their mean values.

*2) **Forest Indices Averaging and Clustering**:* Forest indices are filtered by name and date range. If the number of records is small (¡=12), raw values are returned directly. For larger datasets, the indices are grouped into 12 segments using linear spacing and averaged using aggregation:

```
timestamps = np.linspace(0, count - 1, 12,
    dtype=int)
```

*3) **Gap-Filling via Nearest Timestamp Matching**:* To handle missing forest masks, the system retrieves the nearest available mask in time using the absolute timestamp difference:

```
.annotate(diff=Abs(F("timestamp" -
    data["end_date"]))
.order_by("diff").first()
```

This ensures that the most temporally relevant data is used when generating current analyses, even in the absence of perfect date matches.

*4) **Deforestation Detection**:* This algorithm compares two binary forest mask images to detect deforestation by identifying pixels that were forest (255) in the earlier image and cleared (0) in the later one. It then creates an RGBA image where detected deforested areas are highlighted in red ([255, 0, 0, 255]) for visualization.

```
arr1 = np.array(img1)
arr2 = np.array(img2)

result_arr = (((arr1 == 255) & (arr2 ==
    0)).astype(np.uint8)) # Convert to 255 for
    visualization
height, width = result_arr.shape
color_arr = np.zeros((height, width, 4),
    dtype=np.uint8)

# Where deforestation is detected (mask > 0), set
    the pixel to red.
color_arr[result_arr > 0] = [255, 0, 0, 255]
```

This highlights any area that was marked as deforested in either image. The result is visualized as a binary PNG image.

### I. Features

The system offers a range of features tailored to facilitate forest monitoring and research. The full use-case diagram of the system is shown in Figure 8.

***Forest Mask Visualization**:*

- View geospatial forest coverage over time
- Helps identify vegetated areas using satellite-derived masks

***Burned/Deforested Area Detection*:**

- Detect burned or deforested zones with dedicated masks
- Compare different time periods to observe environmental changes

***Time-Series Forest Indices*:**

- Analyze vegetation health over time using indices such as:
  - NDVI (Normalized Difference Vegetation Index)
  - EVI (Enhanced Vegetation Index)
  - NBR (Normalized Burn Ratio)
  - And more...
- Visualize changes in plant density, moisture, and stress levels

***Forest-Focused AI Chatbot*:**

- Ask questions about forest data, trends, or methods
- Get real-time explanations, insights, and data interpretations
- Built with retrieval-augmented generation (RAG) for domain-specific knowledge

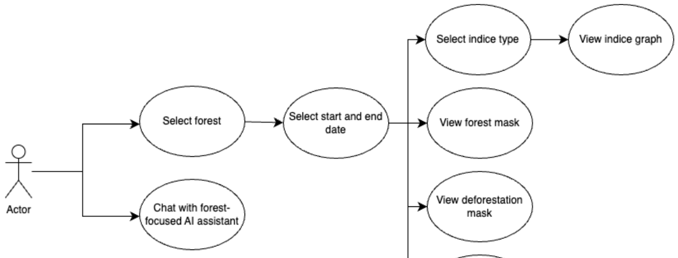


Fig. 8: Use case diagram of the system

### *J. Tools & Third-Party Components*

Our system integrates several powerful third-party tools and services to handle satellite data processing, cloud storage, geospatial analysis, and conversational AI.

- **Google Earth Engine** is used extensively to access, process, and compute spectral indices from satellite imagery, providing a scalable and efficient platform for remote sensing tasks.
- **Google Drive** serves as the cloud storage solution for high-resolution TIFF images and PNG mask files generated during the analysis pipeline.
- **Rasterio** is a Python library used to read, write, and process geospatial raster data, particularly GeoTIFF files. In this project, Rasterio handles the TIFF images from satellite sources, converting them into NumPy arrays for analysis or visualization. It plays a key role in tasks like deforestation detection and mask generation.
- **LangChain** was used to structure prompts and augment responses using external data sources, while the LLM API from **OpenAI** was employed to generate contextually relevant answers. The core backend application is built with **Django and Django REST Framework (DRF)**, providing
- **RESTful** API endpoints and serving as the central integration layer.
- **Postman** was used to test and debug API endpoints, helping ensure smooth data retrieval for forest indices, masks, and chatbot responses during development.
- **Flutter**: Frontend app

## V. Project Execution

### *A. Improvements to the project plan*

Since the beginning of the project, we have been creating an application for forest management leveraging modern AI technologies. The project idea has not changed throughout the past year, but some implementation details have been updated. While researching existing literature and datasets relevant to our project, we noticed that certain aspects of the application development needed modification. The following procedures have undergone some changes:

*1) **Forest dataset creation**:* Initially, the reliance on Sentinel-2 alone led to significant data gaps. Incorporating Landsat imagery and employing gap-filling techniques resolved this issue. The difference in resolution between the two satellite datasets required careful reprojection. Our choice of the nearest neighbor method effectively harmonized the datasets without introducing interpolation errors. We applied an initial automated cloud filtering process with a 20% threshold for Sentinel-2 and Landsat imagery. After this step, we manually verified the images to ensure clarity and quality. However, there are some cloudy pixels in some images. To resolve this issue, we first identify cloudy pixels in the targeted image and then find the nearest clear images taken before and after that date. We compare the classification of the same pixel in both clear images to see if it is marked as a forest. If both clear images indicate forest, we update the cloudy pixel to be classified as forest. This method significantly reduces misclassification and enhances the continuity of our forest mapping.

*2) **Integration of AI into the application**:* The literature and datasets on Kazakhstan's forests are very limited, preventing us from training an AI model to predict forest masks. Nevertheless, we constructed our dataset from scratch using satellite images, mostly Sentinel-2, with its publicly available API and high resolution of images. Also, instead of training an AI model from scratch, we implemented a query-based chatbot to provide insights about forests and answer user questions. This decision helped us widen our project's scope by integrating a real-time AI assistant and our forest maps for an enhanced user experience.

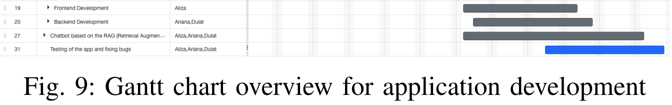
*3) **Chatbot implementation**:* Initially, to create the chatbot's database to retrieve contextual information, we used a Verra project that has a database of project reports on different agricultural and forestry tasks. The preprocessing of all reports took a lot of time, and the quality of the end documents was not as good as expected because of the different structures of the reports. Some were distorted and did not follow document templates, making the data retrieval process difficult. The decision was to create a separate database from scratch consisting of Kazakhstan forest masks and vegetative indices. Later, that dataset was queried to get a relevant context to user questions, which improved the quality of the AI assistant's responses enormously. Also, it allowed us to narrow the scope of Chatbot's contextual database, which helps it provide users with more straightforward and relevant information.

### *B. Team Roles and Responsibilities*

*1) **Initial research of data sources and methodologies**:* Before implementing the application, we had to gather and analyze related literature. First, we found several similar projects, such as CTrees, Pachama, Sylvera, etc. We made a list of those projects and divided them into three parts so that we could separately analyze them and identify their main features, such as datasets used, AI technologies, and satellite imagery. Composing the table with information for each project helped us to determine the sources we need to look into next.

After finding a list of literature, datasets, and satellites from those projects, we again delegated a specific area of research to each team member. For instance, Dulat was making literature summaries, Aliza was learning about available data sources, and Ariana was gathering information about satellites. In the end, each of us presented our findings during weekly meetings, and then we started to work on the dataset creation and implementation of our application. We all chose different things to work on and helped each other during the development process, as shown on the Gantt chart in Figure [**?**]. The Gantt chart shows the overview of all tasks without getting into too much detail for readability, but the more extended version is shown in **Appendix A**.

Every team member was focused on a specific task in different stages of development, but to accelerate and complete the project faster, we also helped each other. For instance, after completing individual tasks, each team member checked and tested them to give valuable feedback and ensure the quality of our product. Also, some tasks were divided into smaller subtasks so everyone could contribute to the project, which also helped us to optimize our resources. The Gantt chart was very crucial for time and resource management during the development process because it helped us to stay on track for the whole project. Even though we did not always strictly follow the plan, we were able to deliver our application components on time.



| ID | Task Name | Resources |
|---|---|---|
| 1 | Sattelite Data Preprocessing | Aliza,Ariana,Dulat |
| 6 | Fire Mapping | Aliza |
| 10 | Forest Mapping | Dulat |
| 14 | Deforestation Mapping | Ariana |
| 18 | Mobile Application | Aliza,Ariana,Dulat |
| 19 | Frontend Development | Aliza |
| 20 | Backend Development | Ariana,Dulat |
| 27 | Chatbot based on the RAG (Retrieval Augmen… | Aliza,Ariana,Dulat |
| 31 | Testing of the app and fixing bugs | Aliza,Ariana,Dulat |

Fig. 9: Gantt chart overview for application development

*2) **Dataset creation and processing**:* Exploring existing datasets showed no data on forests in Kazakhstan, so we needed to find another way to find relevant information. We tried satellite imagery and experimented with available API endpoints for databases of Sentinel-2, MODIS, and LANDSAT satellites. The dataset creation process was organized by assigning clear responsibilities for each workflow step. Responsibilities included sourcing and preprocessing imagery from Sentinel-2 and LANDSAT—applying reprojection, running an automated cloud filtering process with a 20% threshold, and preparing the images for further analysis. Additionally, quality assurance tasks ensured that each image was manually verified and that gap-filling techniques, which compared cloudy pixels to the nearest clear images before and after the target date, were properly implemented to update forest classifications. Then, we downloaded images for selected dates to Google Drive. The link of each image is then stored in the database.

*3) **Chatbot development**:* The plan was to build a database with literature on forestry and agriculture from a wide database of relevant projects called Verra. Verra consists of reports and geo-data from multiple regions worldwide that we wanted to use for the RAG system's database. RAG stands for "Retrieval Augmented Generation," and it helps retrieve a relevant context in the form of text from the database and attach it to the prompts that are later sent to LLM. We built a custom script to scrape and download data from Verra from various forest projects. Then, we spent a decent amount of time preprocessing all information from Verra reports, but it was not ideal due to differences in the files' structures. Most of the reports did not follow templates from Verra, especially old ones, because the templates were changing every year, which made it complicated to preprocess all of them for effective information retrieval from the database. After testing a demo version of RAG using Verra reports, we understood that the system has no context about Kazakhstan's forests. Also, it does not as effectively add context to the prompt as we wanted it to. Therefore, we decided to implement using our forest masks and vegetative indices dataset. The new implementation retrieves data from our database, then does an additional analysis via custom Python scripts, and in the end, answers the user's questions using LLM API endpoints. This approach enhanced the context retrieval part for user prompts and helped improve the quality of the answers provided by the AI assistant. The chatbot was tested on sample questions regarding forest vegetation, burned areas of Semey Ormany forest, deforestation, and various questions about forest management.

*4) **Application development**:* The backend logic was implemented in Python using the Django framework, which closely interacts with the dataset of satellite imagery for specific forests in Kazakhstan. All team members collaborated to share knowledge about different components of our product, including describing the dataset, to accelerate the overall work. All images and indices from the dataset were processed and saved in an SQLite database using Django ORM and custom Python scripts. Storing all data in the local database allows us to retrieve it more quickly and effectively, improving the application's response times.

Then, we implemented backend logic for each type of map and statistical analysis that we incorporated into the application, including forest masks, vegetative indices and graphs, burned areas, and deforestation masks. Everyone tested the backend logic using the Postman application, which allows sending HTTP requests to the local server and viewing the output without the need for the frontend, making it a great tool for the early stages of software development.

In addition, the chatbot backend logic was implemented using Django and was tested using the same techniques as other parts of the code. After finishing the application's internal logic, we developed the visual part, specifically all frontend components, using Flutter. For the visual part, we followed the mobile application design we created in Figma and ensured that all parts were functional and intuitive.

Overall, the process was complicated at times, but through mutual support and collaboration, we successfully implemented a fully functional mobile application.

## VI. Evaluation

### *A. Dataset evaluation*

We evaluated our forest mask classifications by comparing them with visual data from Google Earth. First, we identified specific locations within our study area using the coordinates defined by our forest masks. We then utilized Google Earth Pro's historical imagery feature to obtain satellite images that matched the acquisition dates of our Sentinel-2 and LANDSAT data. Next, we visually inspected areas classified as forests in our masks to confirm that they corresponded to dense vegetation and clear spatial patterns in the Google Earth imagery. Additionally, we verified regions marked as cloud or water, ensuring that these artifacts were correctly identified and not misclassified as deforested areas. Overall, our evaluation demonstrated a high degree of consistency between the forest masks and the visual reference data. The NDVI-based classification method effectively delineated forest from non-forest regions, and our gap-filling strategy maintained the continuity of the mapped data despite minor discrepancies due to noise in data or resolution differences.

### *B. System Evaluation*

*1) **User testing**:* For the user testing, we asked several of our fellow computer science students and acquaintances from forest and agriculture-related projects to asses our mobile application. The total amount of users testing the product is 9. Each participant gave feedback after using all functionalities of the application, such as identifying deforestation in a specific region, checking burned areas, or asking the chatbot a question about forest and vegetative indices. All users commented on the application by saying that the system is very intuitive and it is easy to use. In addition, they noted that the interface is very user-friendly and convenient for any user. The overall design and responsiveness of the application were also given a high score. The general appearance of the mobile application is shown in **Appendix B**. For quantitative results, we asked users for short feedback based on the System Usability Scale (SUS). This scale involves evaluating several statements from 1 to 5 about the product. The statements we used are as follows:

1) I found the forest monitoring app easy to use.
2) The functions and features in the app were well integrated.
3) I felt confident using the system without needing help.
4) The information and visuals (like masks and graphs) were clear and understandable.
5) I would use this kind of app in real-world forest or environmental monitoring.

The SUS scores are visualized using a column chart showing the scores given by each user in Figure [**?**]. The average score is 86.6 out of 100, which means that the application was well-received and easy to navigate. The users scored primarily 4 or 5 for the application's visuals, such as forest, deforestation, and burn maps for the Kazakhstan forest. Additionally, most of them stated that the calculation of vegetative indices is informative and descriptive for assessment of the forest health and the best part is that the application allows a visualization of those indices in the form of graphs within the application. Both types of users familiar and unfamiliar with forest management gave high scores for the usefulness of the chatbot as it provides insights into forests relying on the existing data from our database, and it is handy in cases when a user needs an explanation of some patterns in different index changes and overall about the forest monitoring tasks.

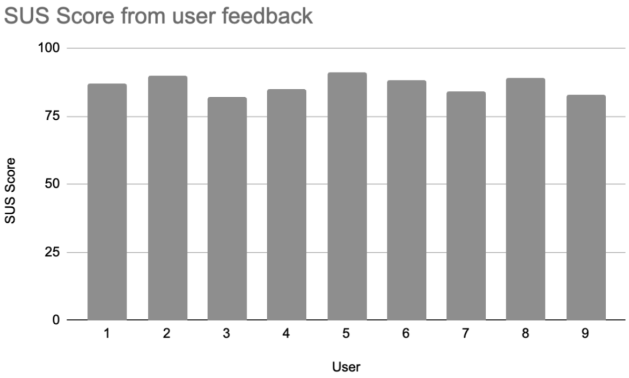


Fig. 10: Column chart showing SUS scores from user feedback

*2) **System Performance**:* We also tested the system's performance on the Android mobile device using an emulator on a Windows machine. We tested the responsiveness of the application ourselves. We noticed that most of the pages are loaded within 2 to 5 seconds which is a decent performance for modern applications. It means that vegetative indices and forest masks are effectively retrieved from the database and shown on the frontend of the application. The chatbot does not take long to respond (around 4 seconds) even though it analyzes the question, retrieves additional information from the database, and only then provides a response to the question, which is a lot of steps. One of the most important parts of the application is deforestation calculations and visualizations on the map, and it takes longer than other pages somewhere around 7 to 10 seconds. It happens because deforestation calculation requires time to compare individual pixels of high-resolution forest cover masks to provide the final result. However, it is not critical as the overall response times show that the backend and database are optimized for smooth and fast usage, even with a lightweight setup.

These evaluations of user testing and system performance show that our forest monitoring application works reliably and is easy to use. The high usability score and responsive system design support that our solution can be useful in real-world scenarios for forest research and environmental management.

## VII. Conclusion and possible future work

Our work presents a comprehensive, automated forest management system that leverages high-resolution Sentinel-2 imagery supplemented by LANDSAT data to monitor Kazakhstan's forests reliably. Key contributions include the creation of a region-specific, biweekly dataset that employs an NDVI-based thresholding method (0.2–0.3) to accurately classify forested areas, alongside robust strategies for cloud filtering, gap filling through temporal interpolation, and careful reprojection of LANDSAT imagery to align with Sentinel-2's resolution. Additionally, our approach integrates a real-time, RAG-based AI chatbot that enhances user engagement by efficiently retrieving and contextualizing data from our system, thus catering to diverse stakeholders. Overall, the project bridges data gaps, improves the precision of forest health monitoring and deforestation detection, and sets a scalable foundation for future advancements in remote sensing and forest management technologies.

For future works, we plan to extend our current monitoring system to cover the entire territory of Kazakhstan, integrating a broader geographic scope of forested areas. Incorporating local weather data such as temperature, precipitation, and humidity will further enhance our analysis by linking meteorological conditions to forest health and disturbances. Additionally, we aim to develop advanced predictive analysis models that forecast deforestation events, fire risks, and other critical changes in vegetation dynamics. This comprehensive approach will improve the accuracy and robustness of forest monitoring and provide proactive, actionable insights for sustainable forest management and climate adaptation strategies.

## References

[1] C. H. L. Silva Junior, V. H. A. Heinrich, A. T. G. Freire *et al.*, "Benchmark maps of 33 years of secondary forest age for Brazil," *Scientific Data*, vol. 7, no. 269, 2020. [Online]. Available: https://doi.org/10.1038/s41597-020-00600-4

[2] F. H. Wagner *et al.*, "Mapping Tropical Forest Cover and Deforestation with Planet NICFI Satellite Images and Deep Learning in Mato Grosso State (Brazil) from 2015 to 2021," *Remote Sens.*, vol. 15, no. 521, pp. 1–21, Jan. 2023. [Online]. Available: https://doi.org/10.3390/rs15020521

[3] F. H. Wagner, R. Dalagnol, and C. H. L. Silva-Junior *et al.*, "Mapping Tropical Forest Cover and Deforestation...," *Remote Sens.*, vol. 15, no. 521, pp. 1–21, Jan. 2023. [Online]. Available: https://doi.org/10.3390/rs15020521

[4] M. Santoro, O. Cartus, and N. Carvalhais *et al.*, "The global forest above-ground biomass pool for 2010...," *Earth Syst. Sci. Data*, vol. 13, pp. 3927–3950, 2021. [Online]. Available: https://doi.org/10.5194/essd-13-3927-2021

[5] P. Rahimzadeh-Bajgiran, C. Hennigar, and A. Weiskittel, "Forest Potential Productivity Mapping...," *Remote Sens.*, vol. 12, no. 2056, pp. 1–17, Jun. 2020. [Online]. Available: https://doi.org/10.3390/rs12122056

[6] J. P. M. Silva, M. L. M. da Silva, and A. R. de Mendonça *et al.*, "Prognosis of forest production using machine learning techniques," *Inf. Process. Agric.*, vol. 10, no. 1, pp. 71–84, 2023. [Online]. Available: https://doi.org/10.1016/j.inpa.2021.09.004

[7] S. Fiandino, J. Plevich, and J. Tarico *et al.*, "Modeling forest site productivity...," *Ann. For. Sci.*, vol. 77, no. 95, pp. 1–9, Oct. 2020. [Online]. Available: https://doi.org/10.1007/s13595-020-01006-3

[8] J. Liu, C. Yue, and C. Pei, "Prediction of Regional Forest Biomass Using Machine Learning: A Case Study of Beijing, China," *Forests*, vol. 14, no. 1008, pp. 1–19, May 2023. [Online]. Available: https://doi.org/10.3390/f14051008

[9] J.-D. Bontemps and O. Bouriaud, "Predictive approaches to forest site productivity...," *Forestry*, vol. 87, pp. 109–128, Nov. 2013. [Online]. Available: https://doi.org/10.1093/forestry/cpt034

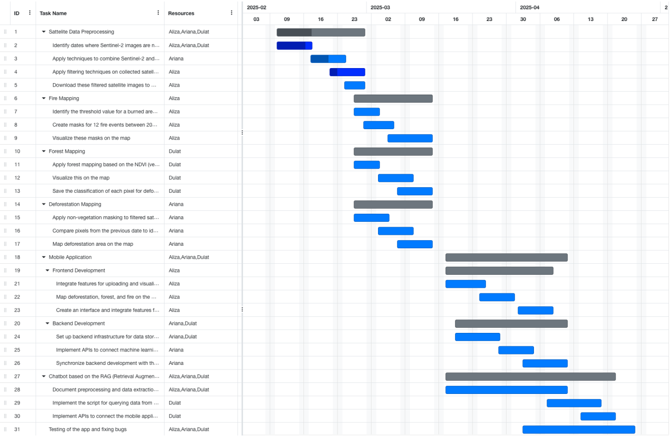
[10] D. Schepaschenko *et al.*, "A dataset of forest biomass structure for Eurasia," *Scientific Data*, vol. 4, no. 170070, pp. 1–10, May 2017. [Online]. Available: https://doi.org/10.1038/sdata.2017.70

[11] L. N. Sotomayor, M. J. Cracknell, and R. Musk, "Supervised machine learning for predicting and interpreting...," *Comput. Electron. Agric.*, vol. 209, no. 107804, pp. 1–14, Apr. 2023. [Online]. Available: https://doi.org/10.1016/j.compag.2023.107804

[12] J. S. Estrada, A. Fuentes, and P. Reszka *et al.*, "Machine learning assisted remote forestry health assessment...," *Front. Plant Sci.*, vol. 14, no. 1139232, pp. 1–15, Jun. 2023. [Online]. Available: https://doi.org/10.3389/fpls.2023.1139232

# Appendix A
## Gantt Chart

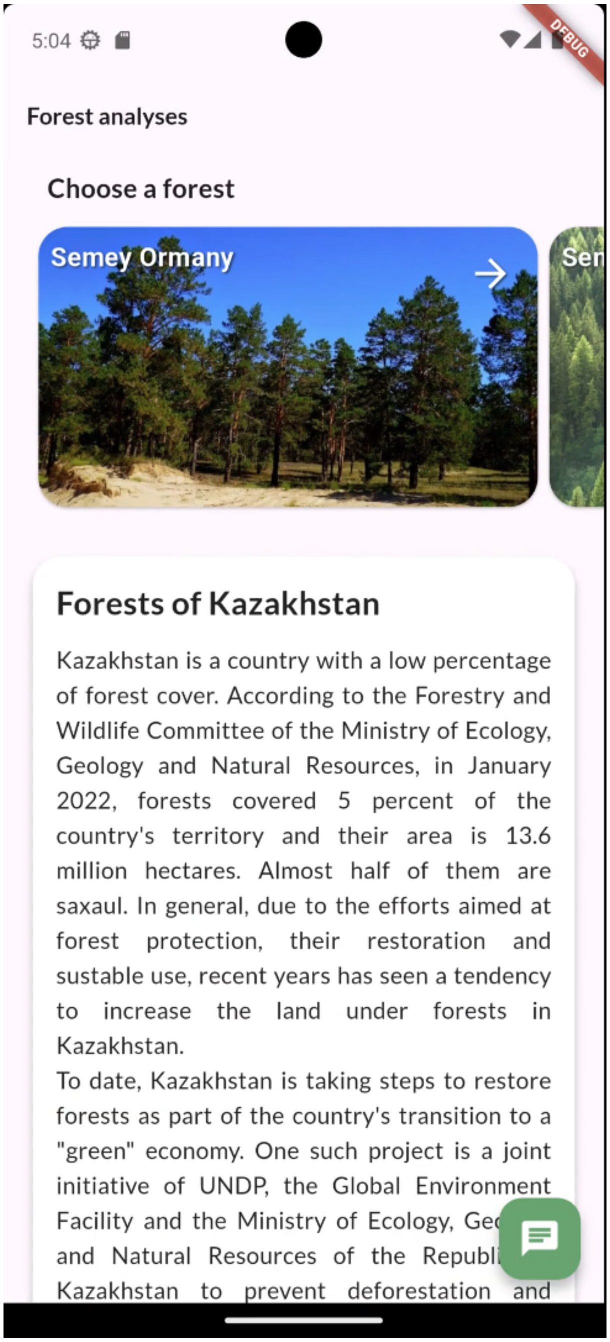
A detailed view of the Gantt chart for time and resource management is needed for the development of the forest management application.



| ID | Task Name | Resources |
|---|---|---|
| 1 | ▾ Sattelite Data Preprocessing | Aliza,Ariana,Dulat |
| 2 | Identify dates where Sentinel-2 images are n… | Aliza,Ariana,Dulat |
| 3 | Apply techniques to combine Sentinel-2 and… | Ariana |
| 4 | Apply filtering techniques on collected satell… | Aliza |
| 5 | Download these filtered satellite images to … | Aliza |
| 6 | ▾ Fire Mapping | Aliza |
| 7 | Identify the threshold value for a burned are… | Aliza |
| 8 | Create masks for 12 fire events between 20… | Aliza |
| 9 | Visualize these masks on the map | Aliza |
| 10 | ▾ Forest Mapping | Dulat |
| 11 | Apply forest mapping based on the NDVI (ve… | Dulat |
| 12 | Visualize this on the map | Dulat |
| 13 | Save the classification of each pixel for defo… | Dulat |
| 14 | ▾ Deforestation Mapping | Ariana |
| 15 | Apply non-vegetation masking to filtered sat… | Ariana |
| 16 | Compare pixels from the previous date to id… | Ariana |
| 17 | Map deforestation area on the map | Ariana |
| 18 | ▾ Mobile Application | Aliza,Ariana,Dulat |
| 19 | ▾ Frontend Development | Aliza |
| 21 | Integrate features for uploading and visuali… | Aliza |
| 22 | Map deforestation, forest, and fire on the … | Aliza |
| 23 | Create an interface and integrate features f… | Aliza |
| 20 | ▾ Backend Development | Ariana,Dulat |
| 24 | Set up backend infrastructure for data stor… | Ariana,Dulat |
| 25 | Implement APIs to connect machine learni… | Ariana |
| 26 | Synchronize backend development with th… | Ariana |
| 27 | ▾ Chatbot based on the RAG (Retrieval Augmen… | Aliza,Ariana,Dulat |
| 28 | Document preprocessing and data extractio… | Aliza,Ariana,Dulat |
| 29 | Implement the script for querying data from … | Dulat |
| 30 | Implement APIs to connect the mobile appli… | Dulat |
| 31 | Testing of the app and fixing bugs | Aliza,Ariana,Dulat |

# Appendix B
## User Interface

Screenshots from the UI of the mobile application. (a) The main page with information on the available forests and context on the forests of Kazakhstan. (b) The interface of the chatbot assistant. (c) Vegetative indices analysis. (d) Deforestation mask mapping. (e) Forest cover mapping. (f) Burn is the mapping of a forest.



Forest analyses

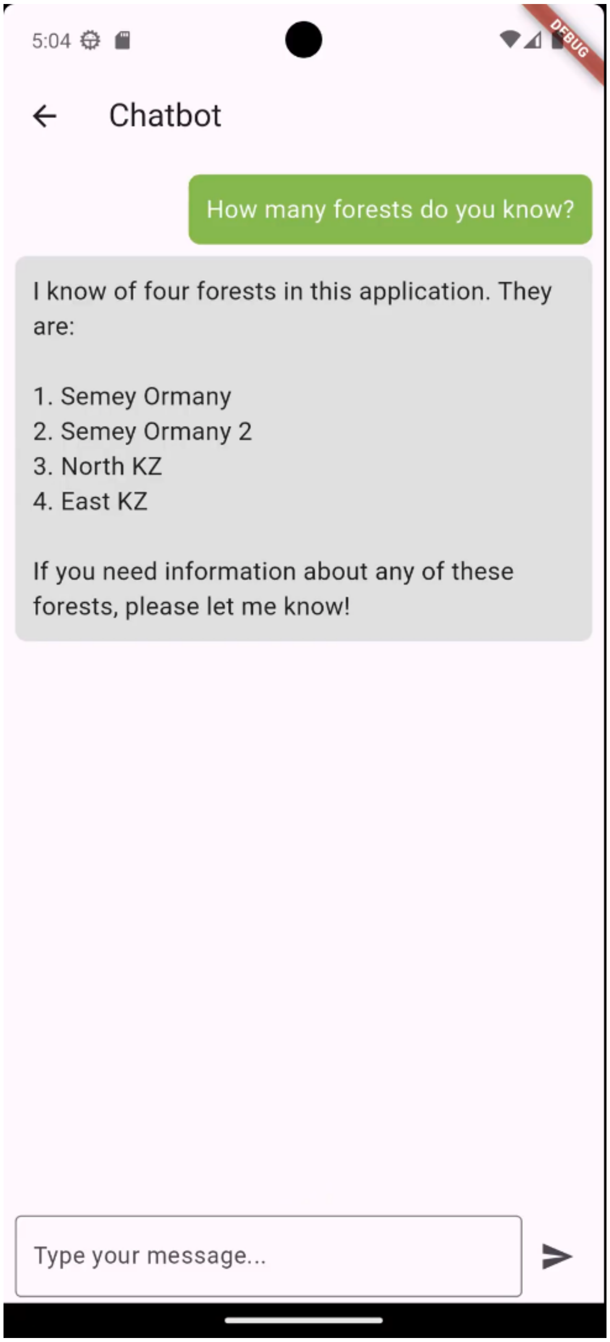
Choose a forest

Forests of Kazakhstan

Kazakhstan is a country with a low percentage of forest cover. According to the Forestry and Wildlife Committee of the Ministry of Ecology, Geology and Natural Resources, in January 2022, forests covered 5 percent of the country's territory and their area is 13.6 million hectares. Almost half of them are saxaul. In general, due to the efforts aimed at forest protection, their restoration and sustable use, recent years has seen a tendency to increase the land under forests in Kazakhstan.

To date, Kazakhstan is taking steps to restore forests as part of the country's transition to a "green" economy. One such project is a joint initiative of UNDP, the Global Environment Facility and the Ministry of Ecology, Geo and Natural Resources of the Republi Kazakhstan to prevent deforestation and

(a)



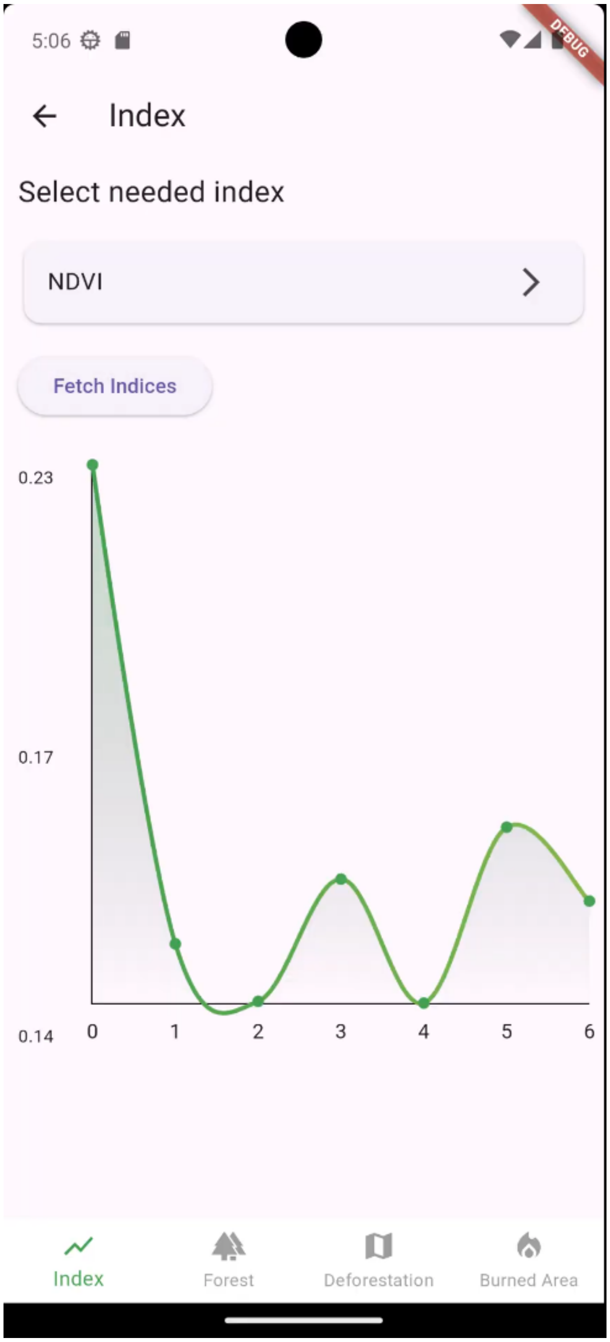
How many forests do you know?

I know of four forests in this application. They are:

1. Semey Ormany
2. Semey Ormany 2
3. North KZ
4. East KZ

If you need information about any of these forests, please let me know!

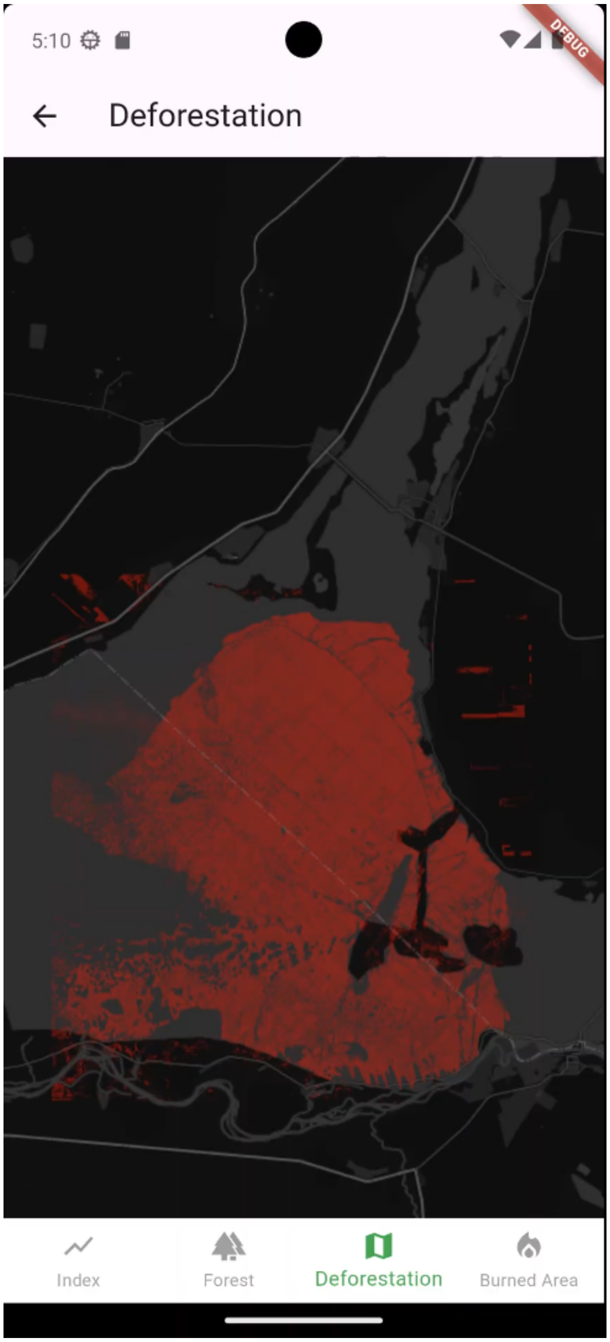
(b)



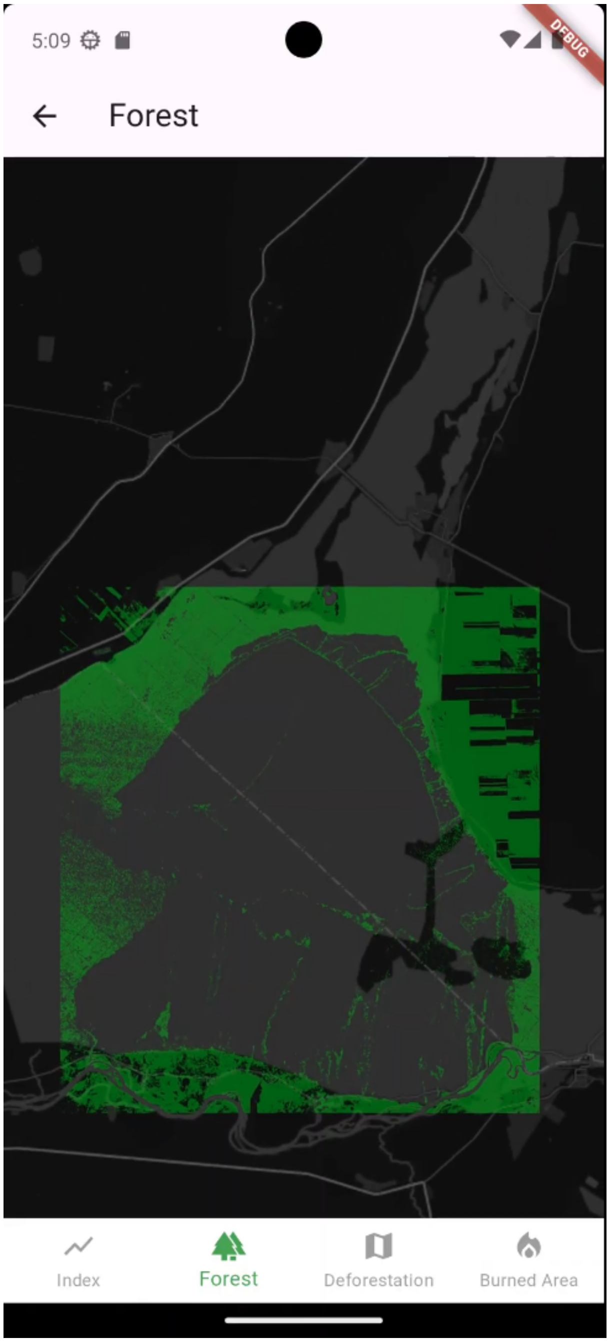
Select needed index

NDVI

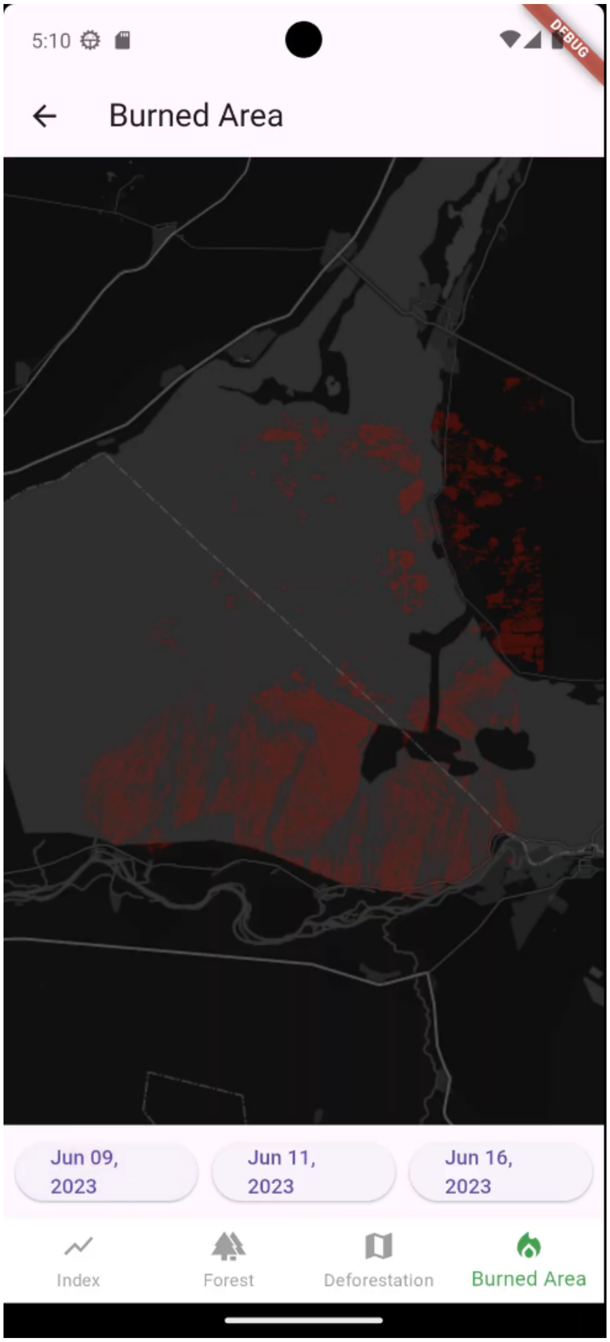
(c)


5:10
DEBUG
Deforestation
Index
Forest
Deforestation
Burned Area

(d)


5:09
DEBUG
Forest
Index
Forest
Deforestation
Burned Area

(e)


5:10
DEBUG
Burned Area
Jun 09, 2023
Jun 11, 2023
Jun 16, 2023
Index
Forest
Deforestation
Burned Area

(f)