Project Documentation: Land Type Classification using Sentinel-2 Satellite Images

Team Members:

١-محمد ايمن فتحي عبد الغني

٢- محمد مراد

٣-يوسف العدوي

٤- محمود محمد محمود

1. Project Planning & Management

1.1 Project Proposal

Problem Statement

Egypt is facing rapid agricultural land loss due to urban expansion, climate change, and water scarcity. Currently, there is no automated, AI-driven system to continuously monitor and predict land-use changes using satellite imagery. This lack of monitoring tools critically threatens food security and sustainable development in line with national vision goals(2030 vision).

Motivation

This project supports Egypt Vision 2030 by promoting sustainable agriculture and environmental monitoring. It will help policymakers track critical land use changes (agriculture, urban, water) and provide farmers and planners with data-driven insights. The goal is to build a high-impact AI and Remote Sensing solution tailored for Egyptian decision-making.

Objectives

1. **Land Classification:** Use Sentinel-2 imagery and advanced AI models (CNN) to classify Egypt's land into Agriculture, Urban, Water, and Desert.
2. **Change Detection:** Monitor and visualize land-use changes from 2015–2023 using Copernicus/WorldCover data (as a baseline for change).
3. **Forecasting:** Predict agricultural land expansion/shrinkage for the next 5 years using time-series models.
4. **Dashboard Development:** Build an interactive platform with maps, charts, and forecasts for decision-makers.

Scope:

1-Data Source: Sentinel-2 RGB satellite imagery (27,000 labeled images):

- **Image Resolution:** 64x64 pixels (RGB)
- **Classes (10):** Forest, Residential, Herbaceous Vegetation, Sea Lake, Annual Crop, Industrial, River, Highway, Permanent Crop, Pasture
- **Geographical Coverage:** African regions, adaptable for Egypt's Nile Delta.
- **System Outputs:**
    - Land-type prediction and confidence score.
    - Interactive map visualization.
    - Real-time information retrieval (SerpAPI).
- **End Users:** Researchers, environmental analysts, policymakers, and students.

Project Plan (Timeline, Milestones & Deliverables)

ID	Phase/Milestone	Deliverable	Deadline	Status
P1	**Phase 1: Research & Setup**	*Completed Documentation (this file)*	**9/21/2025**	To Do
P1.1	Data Acquisition & Preprocessing	EuroSAT (RGB) dataset download and initial cleanup.	9/15/2025	To Do
P1.2	Literature Review Completion	Survey of LULC papers and model architecture comparison.	9/21/2025	To Do
P2	**Phase 2: Model Implementation & Training**	Working classification model and change detection script.	11/15/2025	To Do
P2.1	Baseline CNN Model	Train and test initial CNN on EuroSAT	10/15/2025	To Do
P2.2	Advanced U-Net Model	Implement and train U−Net model for pixel-level classification.	10/30/2025	To Do
P2.3	Time-Series Forecasting	Implement Prophet/ARIMA model for 5-year prediction.	11/10/2025	To Do
P3	**Phase 3: Integration & Finalization**	Interactive Dashboard and Final Report/Presentation.	11/20/2025	To Do
P3.1	Dashboard Prototype	UI/UX implementation and data visualization integration.	11/15/2025	To Do
P3.2	Testing & Validation	Full system testing and final performance metrics.	11/18/2025	To Do

Milestones:

1. Project Proposal Approved
2. Dataset Prepared and Preprocessed
3. CNN Model Trained & Evaluated
4. Flask App Integrated with Model
5. Final Report & Presentation Delivered.

Deliverables:

1. Trained CNN model file (`.h5`)
2. Web application (Flask-based)
3. Prediction & visualization dashboard
4. Documentation (Proposal, Design, Final Report)
5. Presentation slides and demo video

Resource Allocation:

Resource	Purpose	Tool / Platform
Hardware	Model training	Google Colab (GPU)
Dataset	Sentinel-2 images	Kaggle
Software Tools	Development	Python, TensorFlow, Flask
API Integration	Search automation	SerpAPI
Collaboration	Team coordination	GitHub, Google Drive

1.3 Task Assignment & Roles

Team Member	Role	Responsibilities
Mohamed Ayman Fathy	**Data Analyst**.	EDA, visualization ,Oversight, Data preprocessing.
Mohamed Mourad	**Data Scientist**.	Data preprocessing, CNN model training & tuning.
Youssef adawy	AI Engineer	Research validation and review,Evaluate model accuracy.
Mahmoud Mohamed Mahmoud	Web Developer	Flask backend integration, API deployment

1.4 Risk Assessment & Mitigation Plan

Risk	Impact	Likelihood	Mitigation Strategy
Insufficient or unbalanced dataset	High	Medium	Augment dataset, apply normalization
Model overfitting / poor generalization	High	Medium	Use dropout, early stopping, and regularization
Limited computing resources	Medium	High	Use Google Colab GPU or Kaggle resources
API request limits (SerpAPI)	Medium	Low	Cache responses, limit daily calls
Web latency or hosting failure	Medium	Low	Optimize Flask routes, use lightweight models
Data privacy / licensing issues	Low	Low	Use open-source Sentinel data under CC license

1.5 Key Performance Indicators (KPIs)

Metric	Target	Description
Model Accuracy	≥ 90%	Overall classification accuracy on test set
F1-Score	≥ 0.85	Measure of balance between precision & recall
Prediction Time	≤ 2 seconds	Time to classify one image
System Uptime	≥ 95%	Reliability of deployed web app
User Satisfaction	≥ 80%	Feedback from evaluators / users
Data Coverage	100%	All 10 land classes represented

2. Requirements Gathering

2.1 Stakeholder Analysis

Stakeholder	Role / Interest	Needs & Expectations	Impact Level
Environmental Agencies (Egyptian Environmental Affairs Agency)	Monitor deforestation, urban expansion, and land degradation.	Require accurate, up-to-date classification maps for sustainable planning.	**High**
Ministry of Agriculture & Land Reclamation	Agricultural monitoring and land productivity assessment.	Access to crop identification and vegetation trends for resource management.	**High**
Urban Planners / Local Governments	City expansion and zoning optimization.	Visual insights into residential and industrial growth areas.	**Medium**
Researchers & Data Scientists	Academic and environmental research.	Open, reliable data and AI models for further analysis.	**Medium**
NGOs / Disaster Response Teams	Rapid damage assessment post-disasters (e.g., floods, fires).	Quick, reliable detection of affected land types for relief operations.	**Medium**
General Public / Students	Educational and awareness purposes.	Easy-to-use interface for exploring environmental data.	**Low**

2.2 User Stories & Use Cases

User Stories

User	Story
Environmental Analyst	*As an analyst, I want to upload a satellite image so I can automatically classify the land type and detect deforestation.*
Urban Planner	*As a planner, I want to visualize land categories on an interactive map to evaluate urban growth.*
Agriculture Researcher	*As a researcher, I want to analyze vegetation areas to assess farmland conditions and productivity.*
Disaster Response Officer	*As an officer, I want to detect affected zones (flood, fire) from images to guide emergency decisions.*

User	Story
General User	*As a user, I want to understand what type of land an image represents and access related resources.*

Use Cases

Use Case ID	Title	Actors	Description	Pre-Conditions	Post-Conditions
UC-01	Upload Satellite Image	User	The user uploads a Sentinel-2 image to the system.	Image format supported (JPG/PNG).	Image successfully uploaded and processed.
UC-02	Classify Land Type	AI Model	CNN model predicts land category and confidence.	Image preprocessing complete.	Predicted class displayed to user.
UC-03	Display Map Location	System + Leaflet.js	Show image coordinates on interactive map.	GPS metadata available in image.	Map displays location marker.
UC-04	Fetch Related Information	SerpAPI	Retrieve top 3 relevant Google search results for the predicted land type.	Internet connection available.	Resource list displayed on dashboard.
UC-05	View Results Dashboard	User	Visualize prediction, confidence, and map.	Classification complete.	User views results interactively.

2.3 Functional Requirements

ID	Functional Requirement	Priority
FR-01	The system shall allow users to upload Sentinel-2 RGB images.	High
FR-02	The system shall preprocess images for normalization and resizing.	High
FR-03	The system shall classify land type into one of ten categories using CNN.	High
FR-04	The system shall display classification results with confidence percentage.	High
FR-05	The system shall visualize geographic coordinates using Leaflet.js.	Medium
FR-06	The system shall retrieve relevant online information using SerpAPI.	Medium
FR-07	The system shall generate summary statistics and charts for model performance.	Medium
FR-08	The system shall support user-friendly navigation via a web dashboard.	Medium
FR-09	The system shall allow users to download or share classification results.	Low
FR-10	The system shall log all classification activities for testing and evaluation.	Low

2.4 Non-Functional Requirements

Category	Requirement	Description / Standard
Performance	Response Time	Image classification and result display ≤ **2 seconds**.
Accuracy	Model Accuracy	CNN accuracy ≥ **90%** on test data.
Scalability	Model Scalability	System supports future expansion with new image data and model updates.
Security	Data Handling	Only authorized users can upload and access sensitive data.

Category	Requirement	Description / Standard
Usability	User Interface	Simple, responsive, and intuitive dashboard accessible on web browsers.
Reliability	System Uptime	Web application uptime ≥ **95%**.
Maintainability	Code Modularity	Code is well-structured and documented for future updates.
Compatibility	Browser & Device Support	Works on modern browsers (Chrome, Firefox, Edge) and all screen sizes.
Portability	Deployment Flexibility	Flask app deployable on Colab, Streamlit Cloud, or AWS.
Accessibility	Language & Design	English interface with accessible color contrast and readable fonts.

4. System Analysis & Design

4.1 Problem Statement & Objectives

Problem Statement

Environmental monitoring and land-type classification are traditionally manual and time-consuming tasks.
Decision-makers and researchers lack an automated, AI-based tool to:

- Detect land use changes,
- Visualize geographic transformations, and
- Retrieve environmental insights in real-time.

addresses this by combining **AI (CNNs)**, **geospatial visualization**, and **intelligent search integration** to classify land types directly from **Sentinel-2 satellite images** and present results interactively.

Project Objectives

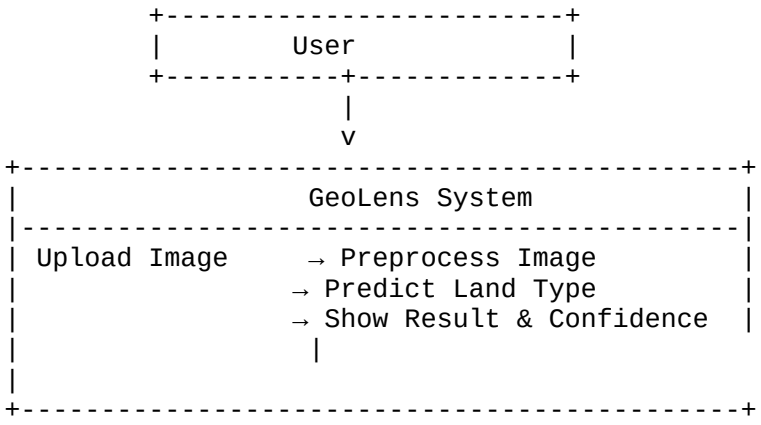
1. Automate land-type classification using deep learning (CNN).
2. Integrate interactive map visualization with GPS-based rendering.
3. Provide real-time contextual data through SerpAPI integration.
4. Build a scalable and user-friendly web dashboard for analysis.
5. Enable model retraining and expansion for future datasets.

4.2 Use Case Diagram & Descriptions

Actors

- **User (Analyst / Researcher):** Uploads images, views results.
- **System:** Performs image preprocessing, classification, and displays outputs.

Use Case Diagram (Description)



Use Case Descriptions

Use Case	Description
Upload Image	User uploads a Sentinel-2 RGB image for analysis.
Classify Land Type	CNN model processes the image and predicts land category.
View Map	Map renders the geographic location of the image.
Fetch Related Info	API retrieves environment-related web resources.
View Dashboard	User sees classification, confidence, and insights.

4.3 Functional & Non-Functional Requirements

These have been described in Section 3 but are summarized here for design reference.

- **Functional:** Image upload, preprocessing, classification, visualization, API integration.
- **Non-Functional:** High accuracy (≥90%), fast response (≤2 s), scalable and secure design.

4.4 Software Architecture

Architecture Style: MVC (Model–View–Controller)

Layer	Components	Description
Model (AI)	CNN model (.h5), data loader, preprocessor	Handles AI logic and classification.
View (Frontend)	HTML, CSS, JS, Leaflet.js	Displays dashboard, map, and results.
Controller (Backend)	Flask app, API routes, SerpAPI integration	Connects model predictions with frontend interface.

4.5 Database Design & Data Modeling

Entities

- **User**
- **Image**

- **Prediction**
- **Location**
- **WebResource**

Entity-Relationship Diagram (ERD)

```
User (user_id, name, email)
   |
   | 1..* uploads
   v
Image (image_id, file_name, upload_date, user_id)
   |
   | 1..1 generates
   v
Prediction (pred_id, class_label, confidence, image_id)
   |
   | 0..1 linked to
   v
Location (loc_id, latitude, longitude, region_name)
   |
   | 0..* linked to
   v
WebResource (res_id, title, url, type, pred_id)
```

Schema Summary

Table	Key Attributes	Notes
Users	user_id (PK), name, email	Stores registered users.
Images	image_id (PK), file_path, upload_date, user_id (FK)	Uploaded Sentinel-2 images.
Predictions	pred_id (PK), label, confidence, image_id (FK)	Model outputs.
Locations	loc_id (PK), lat, lon, region	Geospatial coordinates.
WebResources	res_id (PK), title, url, pred_id (FK)	Related information links.

4.6 Data Flow & System Behavior

Data Flow Diagram (DFD – Level 0)

```
User → [Upload Image] → System → [CNN Model] → [Prediction Results]
                                        ↓
                                  [SerpAPI / Map]
                                        ↓
                                   [Dashboard UI]
```

Sequence Diagram

```
User → Web UI: Upload Image
Web UI → Flask: Send Image
Flask → Model: Predict Class
Model → Flask: Return Result
Flask → SerpAPI: Fetch Info
Flask → Map API: Get Coordinates
Flask → Web UI: Display Output
```

Activity Diagram

```
Start → Upload Image → Preprocess → Classify → Display Results → End
```

State Diagram

State	Description
Uploaded	Image received from user.
Processed	Normalized and resized.
Classified	CNN returns prediction.
Visualized	Displayed with confidence & map.

Class Diagram

```
+------------------+
|  ImageHandler    |
|------------------|
| +upload()        |
| +preprocess()    |
+------------------+

+------------------+
|  CNNModel        |
|------------------|
| +predict(image)  |
+------------------+

+------------------+
|  FlaskController |
|------------------|
| +route_upload()  |
| +route_predict() |
| +route_display() |
+------------------+

+------------------+
|  MapVisualizer   |
|------------------|
| +show_map(lat,lon)|
+------------------+
```

4.7 UI/UX Design & Prototyping

Wireframes & Mockups

Main Pages:

1. **Home Page:** Project description and upload button.
2. **Prediction Dashboard:** Displays predicted land type, confidence score, and class image.
3. **Map View:** Interactive map with location pin (Leaflet.js).
4. **Information Panel:** Links fetched via SerpAPI about the classified region.

4.8 System Deployment & Integration

Technology Stack

Layer	Technology
Frontend	HTML, CSS, JavaScript.
Backend	Flask (Python)

Layer	Technology
AI Model	TensorFlow / Keras
Deployment	Google Colab

Component Diagram

Component	Description
AI Model Component	Performs land-type classification.
Web Interface Component	Collects user input and displays output.
Map Component	Integrates Leaflet.js for visualization.
Search Component	Connects to SerpAPI for online information.
