GreenEye: Project Plan

1. Team Plan for GreenEye Project

The success of the GreenEye project depends on the coordinated efforts of all team members across the various stages of development. Given the project's scale, the team is divided into five specialized roles, each designed to contribute significantly to different aspects of the system while fostering collaboration at key integration points.

Furthermore, the GreenEye project will adopt an **Agile development methodology**. We will operate in **weekly sprints**, with each sprint lasting **one week**. At the beginning of each sprint, a **Sprint Planning meeting** will be held to organize tasks and set sprint goals. At the end of each sprint, a **Sprint Review meeting** will be conducted to demonstrate the completed work, gather feedback, and adjust the next sprint's plans accordingly.

Below is a detailed plan outlining each team member’s role, responsibilities, and how their work fits into the overall timeline.

Team Roles and Responsibilities

Team Member	Role	Main Responsibilities
Majd Al Hatoum	**Requirements Engineer & QA Lead**	- Gather and document functional and non-functional requirements - Define user stories and acceptance criteria - Create detailed system flow diagrams - Design test plans for unit, integration, and user acceptance testing - Lead testing activities and bug reporting
	Data Scientist	- Acquire, clean, and preprocess datasets (Pl@ntNet-300K, PlantVillage, etc.) - Conduct exploratory data analysis - Design and implement data augmentation strategies - Handle class imbalance issues - Prepare datasets for model training and evaluation
	ML Engineer	- Develop and fine-tune deep learning models for species and disease detection - Implement transfer learning using CNN architectures (e.g., ResNet, EfficientNet) - Develop multi-leaf input fusion logic to combine predictions across multiple images - Optimize models for inference speed and memory footprint

Team Member	Role	Main Responsibilities
	Backend Developer (API/Server)	- Document model architectures and training methodologies - Design and implement REST APIs for frontend-backend communication - Integrate ML models into the backend service (e.g., using TensorFlow Serving or custom inference) - Manage user session handling and prediction history storage - Ensure API scalability, security, and performance - Document backend API endpoints (Swagger/OpenAPI)
	Frontend & DevOps Engineer	- Develop a responsive web/mobile application for user interaction - Implement image upload functionality (including multiple images per prediction) - Display model predictions and care suggestions - Set up CI/CD pipelines for automated builds and deployments - Dockerize frontend, backend, and ML services for cloud deployment - Deploy the final system to a cloud provider (AWS, GCP, or Azure)

Work Distribution Principles

This project has been carefully structured to maximize parallel work across team members while ensuring a strong rhythm of collaboration. Most of the project's early tasks — such as requirements gathering, dataset exploration, backend skeleton setup, and initial UI design — are designed to run concurrently. This parallel execution allows the team to make rapid progress from the very beginning, avoiding bottlenecks and ensuring that no one is left waiting for someone else to finish before they can start their work.

At key points during the project, specific handoffs and integrations are planned. For instance, once the Machine Learning Engineer finalizes a baseline model, the Backend Developer will integrate it into the prediction API, and shortly after, the Frontend Developer will consume the API to display results to the user. By clearly defining these handoff moments, we ensure that responsibilities are coordinated and that team members are fully prepared to transition their work to one another smoothly.

In addition to this structured workflow, the team will maintain a high degree of synchronization through the Agile framework. Weekly Sprint Planning and Review meetings will serve as checkpoints for alignment, allowing members to catch integration issues early, adjust priorities based on progress, and improve collaboration across the entire system. Continuous communication, either through Slack or Microsoft Teams, will be encouraged daily to resolve small issues before they become roadblocks.

Finally, while each role has its own specialized focus, cross-functional awareness will be emphasized. Team members are encouraged to stay informed about adjacent activities — for example, the Backend Developer understanding the model's API requirements or the Frontend Developer being aware of backend constraints — to foster a stronger, more resilient integration as the project evolves.

Tools and Technology Stack

- **Machine Learning:** TensorFlow, PyTorch, Keras, Librosa, OpenCV
- **Backend:** FastAPI or Flask, TensorFlow Serving (for model inference)
- **Frontend:** React.js or React Native
- **Database:** MongoDB Atlas or Firebase Realtime Database
- **Deployment:** Docker, Kubernetes (optional), AWS/GCP/Azure
- **Testing:** PyTest (backend and ML), Jest (frontend), Postman (API testing)
- **Project Management:** GitHub Projects, Trello, and Slack for team communication

2. Timeline (Week-by-Week Breakdown)

Week	Main Activities	Lead Roles	Dependencies
Week 1: Planning and Setup	- Requirements gathering - Dataset exploration - UI wireframes and architecture planning - Tech stack setup (GitHub repo, cloud accounts)	- Requirements Engineer (Lead) - Data Scientist - Frontend Dev	None
Week 2: Early Model and API Prototyping	- Build first audio preprocessing pipeline - Train baseline models (small datasets) - Build dummy prediction API - Frontend setup (basic pages)	- ML Engineer - Backend Developer - Frontend Developer	- Dataset access - Wireframes ready
Week 3: Full Development Sprint	- Full model training (Pl@ntNet-300K, PlantVillage) - Build real prediction APIs - Implement multi-leaf input support - Frontend-Backend integration start	- ML Engineer - Backend Developer - Frontend Developer	- Baseline models validated - Backend/Frontend basic structure ready
Week 4: Integration,	- Final integration of	All Members	Model ready

Week	Main Activities	Lead Roles	Dependencies
Testing, and Deployment	frontend, backend, and model - Full system testing (unit, integration, user testing) - Cloud deployment (Docker + cloud hosting) - Prepare final presentation and documentation	QA Lead ensures system tests	- APIs stable - Frontend complete

---

Parallelization Notes

✅ **Weeks 1–2**:

- Requirements Engineer, Data Scientist, Frontend Developer, and Backend Developer can work almost fully in parallel.

✅ **Week 3**:

- ML Engineer finalizes the model while Backend Developer builds the real API, and Frontend Developer connects UI.

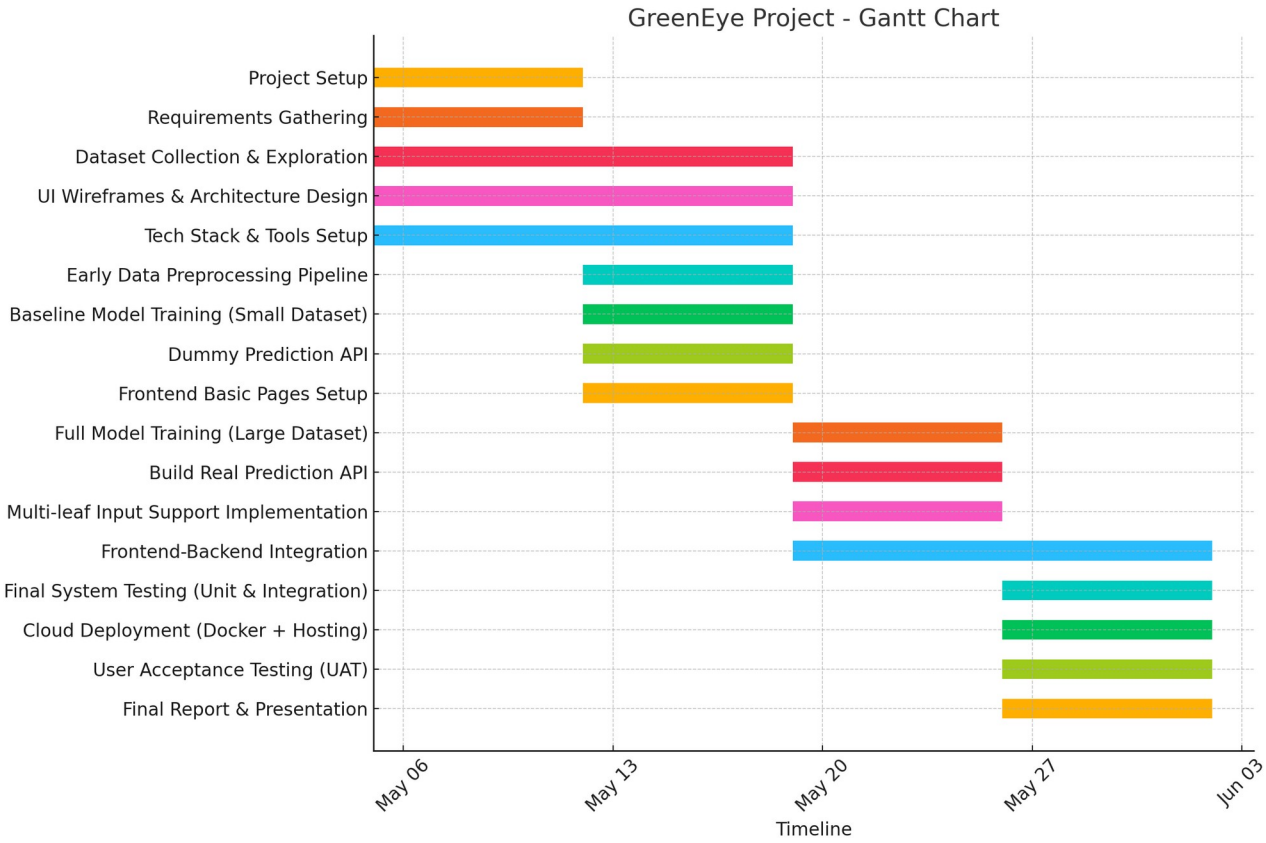
✅ **Week 4**:

- Heavy collaboration week — integration and testing involve everyone. QA Lead drives final bug fixes and testing reports.

3. Gantt Chart

The Gantt chart below provides a visual timeline of all major tasks and activities scheduled across the four-week development period. Each task is plotted against time to illustrate when work begins, when it is expected to end, and how tasks overlap or depend on one another.

The project follows an Agile structure with **weekly sprints**, and tasks are grouped by team roles, including Requirements Engineering, Data Science, Machine Learning, Backend Development, Frontend Development, and Testing & Deployment.



4. Deliverables by End of Project

- Requirements Specification Document
- Well-documented code repositories (ML, API, Frontend)
- Final trained and optimized model
- Dockerized, deployed application (live demo)
- Test reports (unit, integration, usability)
- Final project report & presentation

5. Quick Visual Summary

- **Week 1:** Planning + Setup
- **Week 2:** Baseline Models + Prototypes
- **Week 3:** Full Development + Integration
- **Week 4:** Testing + Deployment + Final Demo