|  |
| --- |
|  |
| Advanced Development System Group Assignment |
| |  |  |  | | --- | --- | --- | | …. | … | … | |

****

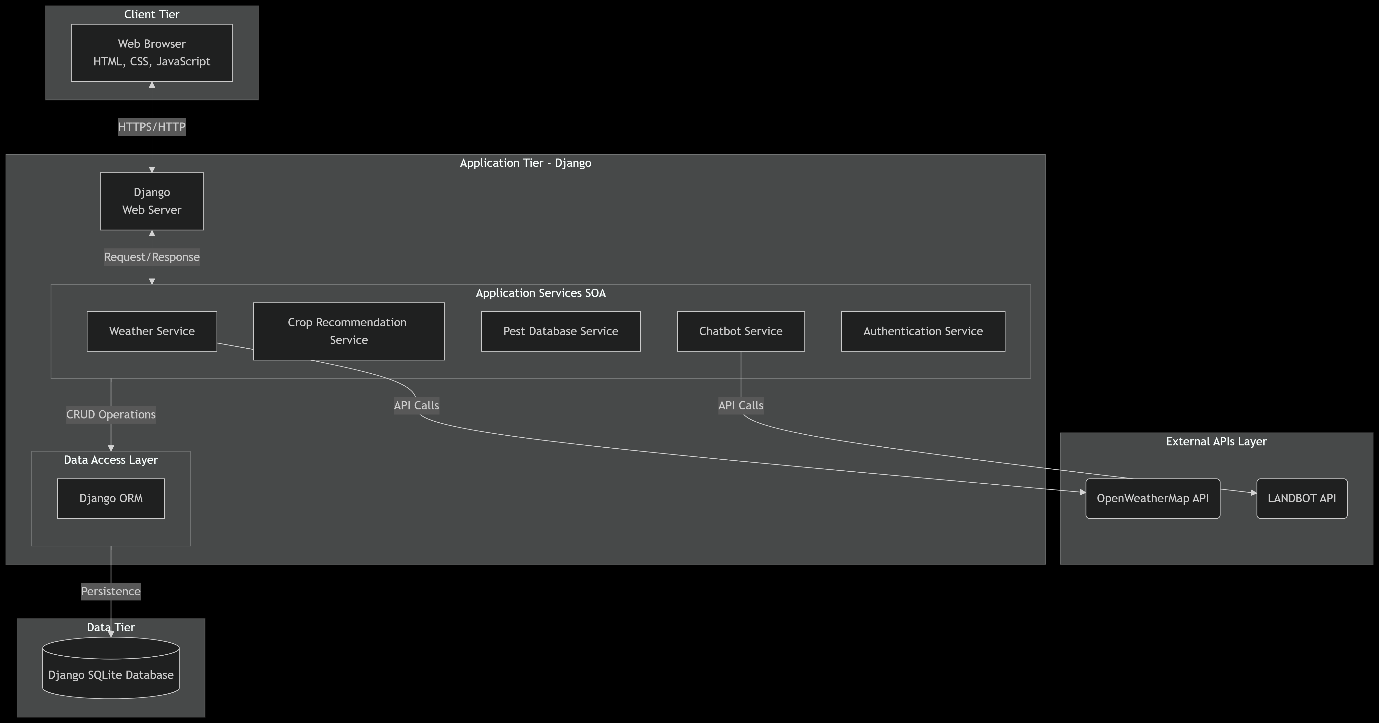
## Group Members

|  |  |
| --- | --- |
| Student Names | Student Number |
| Nkoana Hope Lerato | 202204804 |
| Mawela Mpho Precious | 202233722 |
| Khalo Ayanda Girly | 202213324 |
| Dlamini Kelebogile Sylvia | 202224253 |

****

## System Design

Draw an architecture diagram



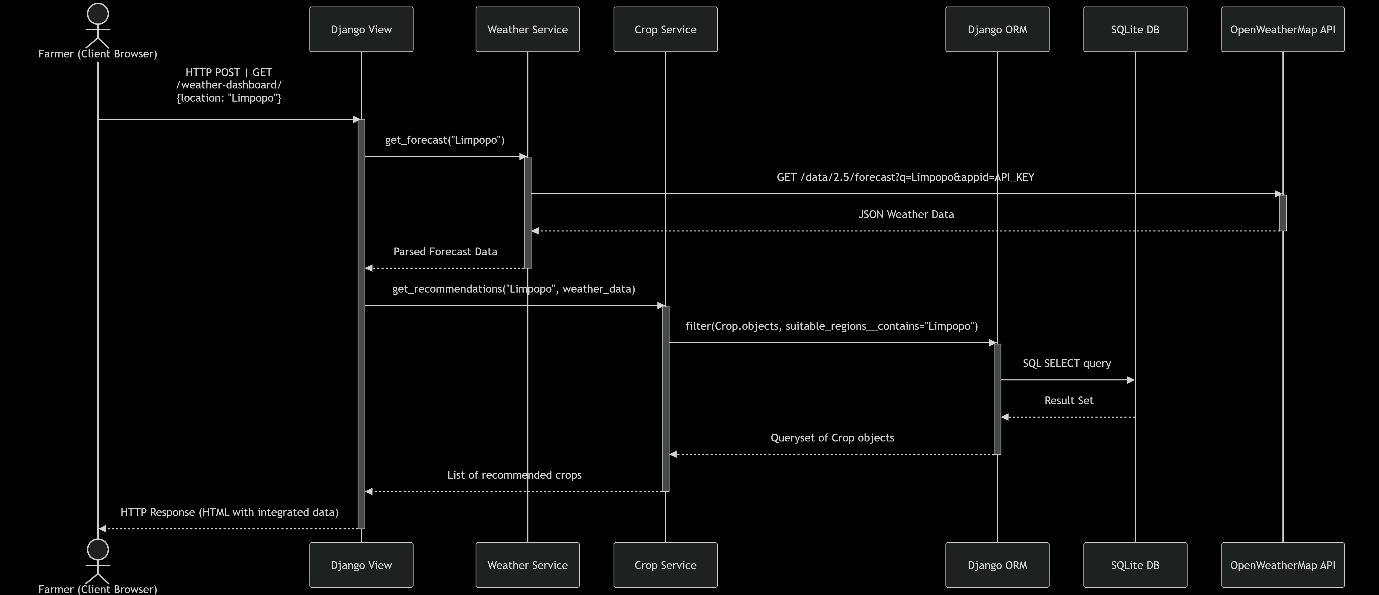
|  |  |  |
| --- | --- | --- |
| **Tier/Layer** | **Component** | **Description** |
| **Client Tier** | **Web Browser** | The user interface rendered on the client's machine. It presents HTML/CSS and uses JavaScript for dynamic interactions and chart rendering. Communicates with the Application Tier via HTTPS. |
| **Application Tier** | **Django Web Server** | The core server that handles HTTP requests, routes them to appropriate views, and returns responses. It serves static files and manages security. |
|  | **Authentication Service** | A dedicated service handling user sign-up, login, session management, and authorization. |
|  | **Weather Service** | A modular service that encapsulates all logic for communicating with the OpenWeatherMap API. It handles API key management, request formatting, and data parsing. |
|  | **Crop Recommendation Service** | Contains the business logic and algorithms for generating crop suggestions and optimal planting calendars based on soil type, location, and processed weather data. |
|  | **Pest Database Service** | Provides an interface to query the internal database for pests relevant to a specific crop and soil type. |
|  | **Chatbot Service** | **This service now specifically integrates with the LANDBOT API.** It formats user queries, sends them to LANDBOT, and processes the responses to display within the application. |
|  | **Django ORM** | The Data Access Layer. Provides an abstraction for communicating with the database, translating Python code into SQL queries. |
| **Data Tier** | **Django SQLite Database** | The primary persistent data store. **This is the default file-based database bundled with Django.** It contains all application tables (Users, Farms, Crops, Pest data, Journal Entries, etc.). It is ideal for development and small-to-medium scale production deployments. |
| **External APIs** | **OpenWeatherMap API** | External REST API providing real-time and forecasted weather data. |
|  | **LANDBOT API** | **The specified external service for chatbot functionality.** This API receives structured or natural language queries from the Chatbot Service and returns AI-generated agricultural advice and responses. |

****

## Data Flow and API Endpoints

**1. Data Flow Sequence Diagram**

The following sequence diagram illustrates the typical data flow for a user requesting a weather forecast and crop advice, showcasing the interaction between all system components.



**2. Internal API Endpoints (Django URLs)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Endpoint (URL Pattern)** | **HTTP Method** | **Purpose** | **Parameters (Request Body/URL)** | **Response** |
| /api/weather/<str:location>/ | GET | Fetches weather data for a given location. | location (URL Parameter) | JSON: { "location": "...", "temp": 25, "forecast": [...], "humidity": 60 } |
| /api/crop/recommend/ | POST | Gets crop recommendations based on location & soil. | JSON: { "location": "...", "soil\_type": "sandy" } | JSON: { "recommendations": [{"name": "Tomato", "reason": "Optimal conditions"}] } |
| /api/pests/<str:crop\_name>/ | GET | Retrieves common pests for a specific crop. | crop\_name (URL Parameter) | JSON: { "crop": "Tomato", "pests": [{"name": "Aphid", "prevention": "..."}] } |
| /api/chatbot/query/ | POST | Sends a user query to the LANDBOT service and returns the answer. | JSON: { "message": "How to treat aphids?" } | JSON: { "response": "You can use neem oil..." } |
| /api/journal/entries/ | GET | Retrieves all journal entries for the logged-in user. | None (uses session) | JSON: [ { "date": "...", "entry": "..." } ] |
| /api/journal/entries/add/ | POST | Creates a new journal entry. | JSON: { "title": "...", "notes": "..." } | JSON: { "status": "Entry saved", "id": 12 } |

**3. External API Integrations**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Service** | **API Endpoint Called (Example)** | **Purpose** | **Authentication** | **Data Sent** | **Data Received** |
| **OpenWeatherMap** | https://api.openweathermap.org/data/2.5/forecast?q={city}&appid={key} | To retrieve 5-day weather forecasts for a user-specified location. | API Key (appid) as a URL parameter. | Location name (e.g., q=Limpopo). | A JSON object containing weather data arrays (list[dt, main.temp, weather[0].description]). |
| **LANDBOT** | https://api.landbot.io/v1/chatbot/{bot\_id}/message | To send user messages and receive AI-generated agricultural advice. | Likely an API Key in the request header (e.g., Authorization: Token {key}). | A structured JSON payload with the user's message and potentially user\_id. | A JSON response containing the chatbot's text response. |

****

## Setup and Tools Used

Setup and Tools Used

|  |  |  |
| --- | --- | --- |
| **Technology / Tool** | **Purpose** | **Version / Notes** |
| **Backend** | **Python** | Core programming language for application logic. | 3.10+ |
|  | **Django** | High-level Python Web framework for rapid development. | 4.2+ |
|  | **Django REST Framework** | Toolkit for building Web APIs (if AJAX-heavy). | 3.14+ |
| **Frontend** | **HTML5** | Markup language for page structure. |  |
|  | **CSS3** | Styling and layout of web pages. |  |
|  | **JavaScript (ES6+)** | Client-side interactivity and dynamic content. |  |
|  | **Bootstrap** | CSS framework for responsive and mobile-first design. | 5.3+ |
|  | **Chart.js** | JavaScript library for rendering weather data charts. | 3.9+ |
| **Database** | **SQLite** | Default lightweight database for development. | Bundled with Django |
| **Development** | **VS Code** | Primary source-code editor. | With Python extension |
|  | **Git** | Distributed version control system. |  |
|  | **GitHub** | Cloud-based hosting for Git repository collaboration. |  |
| **External APIs** | **OpenWeatherMap API** | Provides real-time and forecast weather data. | Free/Paid Tier |
|  | **LANDBOT API** | Provides conversational AI for the chatbot feature. |  |

**2. Development Environment Setup**

**Prerequisites:**

* A computer running Windows
* An internet connection.
* A GitHub account.
* API keys for OpenWeatherMap and LANDBOT.

3. Project Structure

