TerraByte

Technical Installation Manual

A project for EPI-USE® by GeekGurusUnion

Demo 4

30 September 2024

Table Of Contents

Introduction

The AI-Driven Crop Prediction System is a comprehensive solution that applies Machine Learning and AI to analyze weather, soil, and crop data, predicting crop health and yield. The system is divided into two main components: the frontend, built using Nuxt 3, and the backend, powered by Python. To fully configure and run the system, you will need to install several dependencies and configure environment variables. This manual will guide you through the steps to set up the system on a Windows operating system.

User Guide

This section will help users navigate to and use the website effectively.

1. **Open Your Web Browser:** Launch your preferred web browser (e.g., Chrome, Firefox, Safari).
2. **Enter the Website URL:** In the address bar at the top of your browser, type the following URL: https://terrabyte.software
3. **Press Enter:** After typing the URL, press the Enter key on your keyboard to navigate to the website.
4. **Log In (if required):** If authentication is required, enter your username and password. If you don't have an account, follow the sign-up instructions provided on the website.
5. **Navigate the Application:** Once logged in, explore the website's features and functionalities. Use the navigation menu to access different sections.
6. **Get Help:** If you encounter any issues or need assistance, refer to the Help section on the website or contact support at terrabyte.app@gmail.com.

Developer Guide

Prerequisites

The following tools must be installed and configured before you can run the software. You will also need the supabase functions and schema to accurately clone the entire system.

Software Installation Guides

These software installation guides will help you install and configure the required versions. Follow the installation guides based on your operating system:

- **Node.js:** Download and Install Node.js
- **Pnpm:** Download and Install PNPM
- **Python:** Download and Install Python
- **Pip:** Pip installation guide
- **Google Cloud Console:** Create a Project and Get an API Key -> Google Cloud Console
- **Supabase:** Set Up Supabase Project -> Supabase
- **OpenWeather:** Get an OpenWeather API Key -> OpenWeather
- **Resend:** Get an Resend API Key -> Resend
- **Gemini:** Get Google Gemini API Key -> Gemini

General Requirements

Before installing the system, ensure that the following software is correctly installed on your machine:

- **Node.js**: Version **18.18.0**
- **PNPM**: Version **9.4.0** (used for managing frontend dependencies)
- **Python**: Version **3.10.8**
- **Pip**: Version **24.0** (included with Python)

Frontend Prerequisites

For the frontend, ensure you have the following setup:

- **Google Cloud Console:** Create a project and generate a Google Maps API key for map integration.
- **Supabase:** Set up a project to manage your database. Make sure to obtain your project URL and API key from Supabase.
- **Resend:** Create a resend account and generate an API key for the project.

Backend Prerequisites

For the backend:

- **Supabase**: The backend requires the Supabase project URL and API key for database operations.
- **OpenWeather**: Get an API key from OpenWeather to fetch weather data.

Installation

Cloning the Repository

Open the folder where you want to store this project. Then you can start by cloning the main branch of the repository to your local machine:

```
git clone https://github.com/COS301-SE-2024/Crop-Prediction-System.git
```

Frontend Installation

Open the folder where you cloned the repository.

1. Navigate to the Frontend Directory:

```
cd frontend
```

2. Install Frontend Dependencies:

```
pnpm install
```

3. Configure Environment Variables: Create a .env file in the frontend directory and add the following variables:

```
GOOGLE_MAPS_API_KEY = your_google_maps_api_key
SUPABASE_URL = your_supabase_url
SUPABASE_KEY = your_supabase_key
API_BASE_URL = http://localhost:8000
RESEND_API_KEY = your_resend_api_key
APP_BASE_URL = http://localhost:3000
```

Backend Installation

1. Installing pip:

```
python -m ensurepip
python -m pip install --upgrade pip==24.0
```

2. Install Backend Dependencies: Install the required Python packages listed in the requirements.txt file:

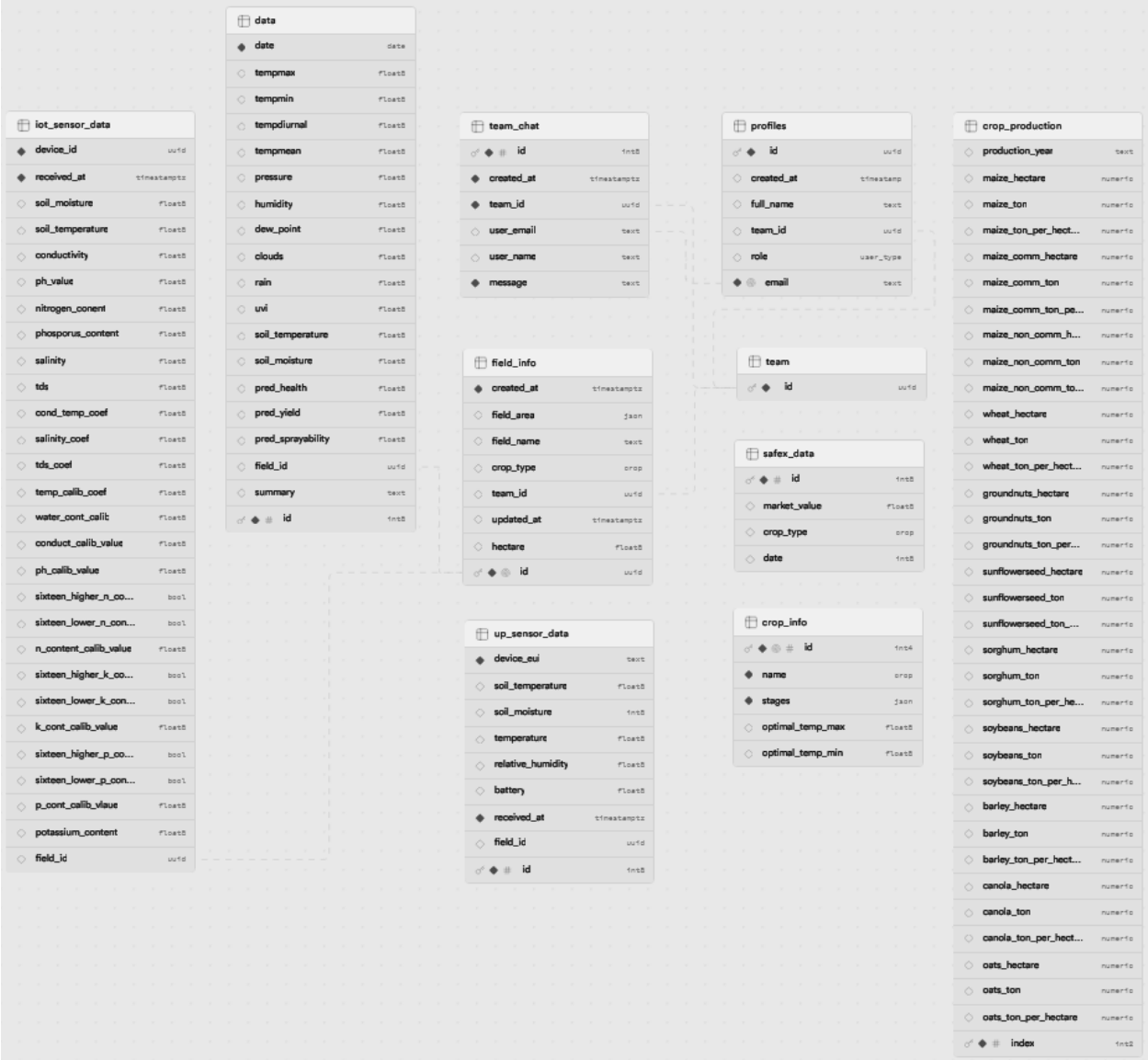
```
pip install -r backend/requirements.txt
pip install uvicorn
```

3. Configure Environment Variables: Create a .env file in the backend directory and add the following variables:

```
SUPABASE_URL = your_supabase_url
SUPABASE_KEY = your_supabase_key
OPENWEATHER_API_KEY = your_openweather_api_key
GEMINI_API_KEY = your_gemini_api_key
MODEL_DIR = /
SENSOR_API_KEY = your_sensors_api_key
SENSOR_API_URL = your_sensors_api_key
```

Supabase Setup

Database Schema



Functions

Our Supabase contains 15 functions which are crucial for a smooth running application. Below are the functions and their definitions:

Function Name	Definition	Arguments
get_all_fields_f	`SELECT * FROM field_info WHERE field_info.team_id`	`teamid : uuid`

rom_team	`= teamid;`	
get_data	`SELECT * FROM data WHERE field_id IS NULL OR field_id = fieldid;`	`fieldid : uuid`
get_data	`SELECT * FROM data WHERE field_id IS NULL;`	
get_field_info	`SELECT * FROM field_info WHERE field_info.id = field_id;`	`field_id : uuid`
get_data_from_team	`SELECT * FROM data INNER JOIN field_info ON data.field_id = field_info.id INNER JOIN team ON field_info.team_id = team.id WHERE team.id = teamid;`	`teamid : uuid`
get_distinct_crop_yield_by_team	`SELECT DISTINCT crop_type, SUM(hectare) AS total_hectare, SUM(fdata.pred_yield) AS pred_tph, SUM(hectare * fdata.pred_yield) AS pred_production FROM field_info LEFT JOIN (SELECT pred_yield, field_id FROM data WHERE date_trunc('day', data.date) = current_date) as fdata ON field_info.id = fdata.field_id WHERE team_id = teamid GROUP BY crop_type;`	`teamid : uuid`
get_field_data_by_id	`SELECT * FROM data WHERE data.field_id = fieldid;`	`fieldid : uuid`
get_historical_yields	`BEGIN RETURN QUERY EXECUTE format('SELECT production_year, %I FROM crop_production WHERE %I IS NOT NULL', crop, crop); END;`	`crop : text`
get_market_value_by_crop	`SELECT date, market_value FROM safex_data WHERE crop_type = croptype;`	`croptype : crop`

get_model_data	`SELECT * FROM data WHERE field_id IS NULL`	
get_model_data	`SELECT * FROM data WHERE field_id IS NULL OR field_id = fieldid;`	`fieldid : uuid`
get_recent_entries	`SELECT * FROM data ORDER BY date DESC LIMIT n`	`n : integer`
get_team_id	`SELECT * FROM profiles WHERE profiles.id=userid;`	`userid : uuid`
handle_new_user	`DECLARE x UUID; BEGIN x = gen_random_uuid(); INSERT INTO public.team (id) VALUES (x); INSERT INTO public.profiles (id, full_name, email, team_id) VALUES ( NEW.id, NEW.raw_user_meta_data ->> 'full_name', COALESCE(NEW.raw_user_meta_data ->> 'email', NEW.email), x );`	
remove_from_team	`DECLARE new_team_id UUID := uuid_generate_v4(); BEGIN INSERT INTO team (id) VALUES (new_team_id); UPDATE profiles SET team_id = new_team_id, role = 'farm_manager' WHERE id = user_id; END;`	`user_id : uuid`

Deployment/Running

Running the Frontend

Change directory to the Frontend Directory and start the Nuxt Development Server:

```
pnpm run dev
```

1. The frontend will be running at http://localhost:3000.

Running the Backend

Start the FastAPI Server:

```
uvicorn backend.app:app --reload --host 127.0.0.1 --port 8000
```

1. The backend will be running at http://localhost:8000.

Running Tests

Frontend Tests

To run the frontend tests:

```
pnpm run test
```

Backend Tests

To run the backend tests:

```
python3 -m pytest
```

Additional Resources and Guidelines

Development Workflow

- **Branching Strategy:** We are using Git Flow as the branching strategy for this project.
- **Code Style Guidelines:** Coding Standards

Documentation and Resources

- Additional Documentation:
  - GitHub Wiki
  - User Manual