



FERTIGuide

(A Fertilizer Recommendation Startup)

(Version 1.0)

Created by:

Deepthi Kumari Rayapureddy - 202419006

Nidhi Chaudhary - 202419004

Copyrights © 2024 by DAIICT, Gandhinagar

TABLE OF CONTENTS

Table of Contents

1. Software Requirement System

- Introduction
- Overall Description
- Specific Requirements
- Logical Data Model
- Data Collection Techniques
- Operating Environment
- User Classes and Characteristics
- Product Functions
- Assumptions and Constraints
- Business Constraints

2. Relational Model Diagram

3. DDL and SQL Queries

- DDL Script
- Plain Text and SQL Queries
- Executed Queries Outputs

Contributions (Team Member-wise)

Overall Take Away from the Course

SOFTWARE REQUIREMENT SPECIFICATIONS

1. INTRODUCTION

Purpose:

The FERTIGuide project aims to create a web-based application that assists farmers in choosing the optimal fertilizer based on soil properties and crop needs. The system will provide a user-friendly platform for real-time, region-specific fertilizer recommendations to improve crop yield and minimize excess fertilizer usage.

Intended Audience:

- **Farmers:** For optimizing fertilizer usage in their crops.
- **Agricultural Experts:** To offer advisory services.
- **Government Organizations:** To improve service delivery related to soil health and crop production.
- **Developers and Testers:** For system implementation and validation.

Objectives:

- Provide accurate, data-driven fertilizer recommendations based on soil and crop data.
- Minimize environmental damage from over-fertilization.
- Enable easy-to-use interfaces accessible via web and mobile.
- Integrate with local and regional databases for soil, weather, and crop data.
- Support agricultural advisory services through expert systems.

Scope:

This project aims to provide a recommendation system accessible through both web and mobile platforms, using machine learning to analyze soil properties (pH, NPK values, etc.), crop type, and weather conditions for optimal fertilizer selection. The FERTIGuide platform will integrate with external soil testing and weather databases to generate personalized recommendations.

2. OVERALL DESCRIPTION

Product Perspective:

FERTIGuide will integrate soil testing data, crop-specific requirements, and regional weather patterns into a recommendation engine powered by machine learning models, ensuring farmers receive real-time, region-specific recommendations.

Product Functions:

- **Soil Data Input:** Enables manual or automated entry of soil health data.
- **Crop Selection:** Users can input their current crop or select from a database of predefined crops.
- **Fertilizer Recommendations:** Based on soil and crop inputs, the system generates fertilizer suggestions.
- **Report Generation:** Users can generate and export a PDF report detailing recommendations.
- **Feedback and Support:** Users can submit feedback or consult with experts for additional guidance.

Business Scenarios:

- **Current State:** Farmers often rely on generalized advice, which can lead to incorrect fertilizer usage.
- **Future State:** FERTIGuide automates the recommendation process, ensuring that every farmer receives data-driven and customized advice.

3. SPECIFIC REQUIREMENTS

External Interface Requirements:

- **User Interfaces:** A responsive web interface with support for low-latency areas; a simplified mobile interface for farmers with limited access to technology.
- **Hardware Interfaces:** Integration with soil testing devices via Bluetooth or USB connections; interfacing with external databases such as Soil Health Card or regional crop yield datasets.
- **Software Interfaces:** Integration with APIs for local weather updates and regional agricultural data; communications interfaces for SMS and email notifications.

Product Features:

1. **Use Case 1: Data Input**
 - Actors: Farmer
 - Pre-condition: Farmer has soil test data or performs the test.
 - Description: The user inputs soil test data (pH, NPK, moisture, etc.) via web/mobile interface.
 - Post-condition: The data is validated and stored for processing.
2. **Use Case 2: Fertilizer Recommendation**
 - Actors: Farmer, Agricultural Expert
 - Pre-condition: Soil and crop data must be available.
 - Description: Based on soil data and crop selection, the system provides fertilizer suggestions.
 - Post-condition: A detailed report is generated, offering optimal fertilizer quantities and application timing.
3. **Use Case 3: Report Generation**
 - Actors: Farmer, Agricultural Expert
 - Description: After the recommendation is generated, a downloadable PDF report is created summarizing the fertilizer application strategy.

4. LOGICAL DATA MODEL

Soil Data:

Soil characteristics play a critical role in determining which fertilizers to use and in what amounts. This dataset typically includes pH level, Nitrogen, Phosphorus, Potassium (NPK), moisture content, and micronutrients.

Crop Data:

Tailoring fertilizer to the specific crop being grown is essential. Key factors include:

- Crop type
- Growth stage
- Desired yield

Different crops require varying levels of nutrients; for example, rice has high nitrogen demands while legumes require less due to their nitrogen-fixing abilities.

Fertilizer Database:

The recommendation system should connect to a comprehensive fertilizer database that includes:

- List of fertilizers
- Chemical composition
- Costs

Each fertilizer's nutrient composition must be factored in to calculate exact application rates for balanced nutrition.

Weather Data:

Weather conditions significantly impact fertilizer effectiveness; therefore, real-time weather data is crucial for recommendations. Weather conditions play a crucial role in determining the effectiveness of fertilizer application. For instance, factors such as temperature, humidity, and rainfall can influence nutrient availability and absorption by crops. Real-time weather data enables farmers to apply fertilizers at optimal times, reducing waste and enhancing crop yield. Additionally, understanding weather patterns helps mitigate environmental impacts, such as runoff or nutrient leaching, ensuring sustainable agricultural practices.

5. DATA COLLECTION TECHNIQUES

Interviews:

Interviews with local farmers revealed varied practices in soil testing and significant challenges in fertilizer use. Interviews conducted with local farmers revealed a diverse range of practices regarding soil testing, highlighting significant challenges faced in fertilizer usage. Many farmers reported relying on traditional methods of soil assessment, such as visual inspections or anecdotal evidence, rather than utilizing scientific soil testing techniques. This reliance often leads to inconsistent application of fertilizers, as farmers may not accurately understand their soil's nutrient needs.

For instance, farmers in regions like Palanpur and Ahmedabad predominantly use broadcasting methods for applying fertilizers like nitrogen (urea) and phosphorus (DAP). However, these methods can result in uneven distribution and inefficient nutrient uptake by crops. Additionally, farmers expressed concerns over the high costs associated with modern soil testing technologies and the lack of accessibility to such services, particularly in remote areas.

Moreover, smaller-scale farmers (those managing less than one acre) emphasized the need for affordable and user-friendly technologies that could facilitate precise fertilizer application. In contrast, larger-scale farmers in areas like Pulivendula had better access to retailers and information regarding fertilizer selection but still faced challenges related to application costs and the effectiveness of their chosen methods.

Overall, these interviews highlighted a critical gap in knowledge and resources that FERTIGuide aims to address by providing data-driven recommendations based on accurate soil health assessments. This approach will empower farmers to make informed decisions that enhance crop productivity while minimizing environmental impacts associated with over-fertilization.

Background Readings:

A review of government guidelines on soil health management reinforced the need for region-specific recommendations due to diverse soil types. A review of government guidelines on soil health management reinforced the necessity for region-specific fertilizer recommendations, particularly due to the diverse soil types found across different agricultural regions. These guidelines emphasize that soil characteristics, such as texture, pH, and nutrient content, vary significantly from one area to another.

For example, sandy soils typically drain quickly and may require more frequent fertilization compared to clay soils, which retain moisture but can become compacted.

Government publications highlight the importance of understanding local soil conditions to optimize fertilizer application. This understanding is crucial because inappropriate fertilizer use can lead to nutrient runoff, soil degradation, and reduced crop yields. By tailoring recommendations to specific soil types, farmers can enhance nutrient efficiency and improve overall agricultural productivity.

Furthermore, the guidelines advocate for the integration of soil testing into regular farming practices. They suggest that farmers should conduct soil tests before planting seasons to accurately assess nutrient needs. This proactive approach allows for adjustments in fertilizer types and quantities based on real-time data rather than relying on generalized advice.

The review also pointed out that existing fertilizer recommendation systems often overlook the variability in factors such as moisture levels and temperature, which can fluctuate significantly even within small geographic areas. By aligning fertilizer recommendations with these localized conditions, farmers can make more informed decisions that lead to sustainable farming practices.

In summary, government guidelines underscore the critical need for region-specific fertilizer recommendations that consider the unique characteristics of local soils. This approach not only supports better crop management but also promotes environmental sustainability by minimizing the risks associated with over-fertilization and nutrient runoff.

Questionnaire:

A Google Form survey was conducted to gather insights from farmers regarding their practices.

1. *Name:*
2. *Location:*
3. *Contact Number:*
4. *What is the size of farm?*
 - a. *< 1 acre*
 - b. *1-5 acres*
 - c. *>5 acres*
5. *Which crops do you grow?*

- a. *Paddy*
 - b. *Wheat*
 - c. *Barley*
 - d. *Maize*
 - e. *Groundnut*
 - f. *Vegetables*
 - g. *Fruits*
 - h. *Other: pls specify*
6. *What type of soil is present in your farm?*
- a. *Sandy*
 - b. *Clay*
 - c. *Loamy*
 - d. *Silt*
 - e. *Mixed*
7. *Which fertilizers do you use mostly?*
- a. *Nitrogen (Urea, etc.)*
 - b. *Phosphorous (DAP, etc.)*
 - c. *Potassium (MOP, etc.)*
 - d. *Organic Manures*
 - e. *Micronutrients (Zn, B, etc.)*
 - f. *Other: pls specify*
8. *How do you apply fertilizers?*
- a. *Broadcasting*
 - b. *Band application*
 - c. *Foliar application*
 - d. *Fertigation*
 - e. *Other: pls specify*
9. *How often do you apply fertilizers?*
- a. *Once per season*
 - b. *Twice per season*
 - c. *> Twice per season*
 - d. *As required*
10. *Rate the water availability in your farm (On a scale of 5)*
11. *What challenges do you face in fertilizer application? (High-cost availability, lack of technology, etc.)*

12. Where do you get information about fertilizers?

- a. Scientists
- b. Other Farmers
- c. Government
- d. Retailers
- e. Online Sources
- f. Other: pls specify

13. Any other suggestions or feedback regarding fertilizer usage?

Observations:

Field observations indicated that while farmers are aware of specific needs, they often lack tools or knowledge for optimal strategies.

Location

25 responses

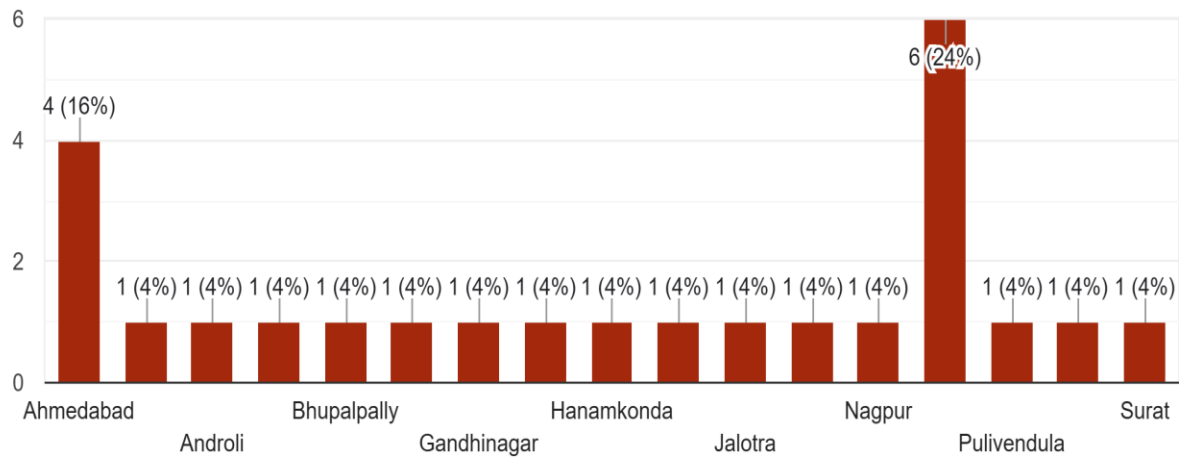


Fig. 1: Location of Responses

What is the size of your farm?

25 responses

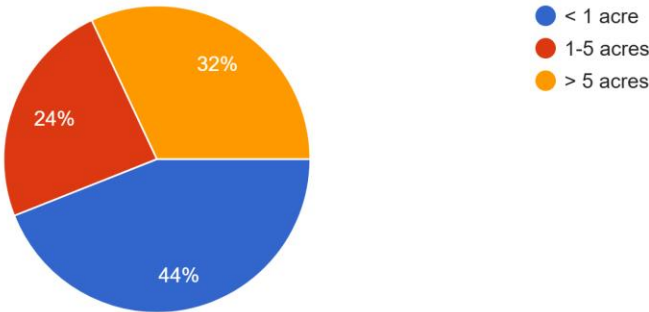


Fig. 2: Size of the Farm

Which crops do you grow?

25 responses

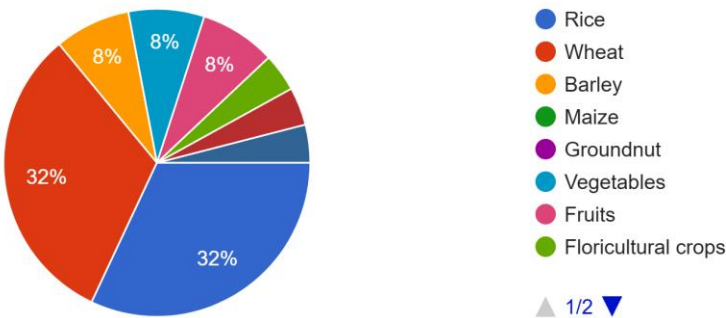


Fig. 3: Crops grown

What type of soil is present in your farm?

25 responses

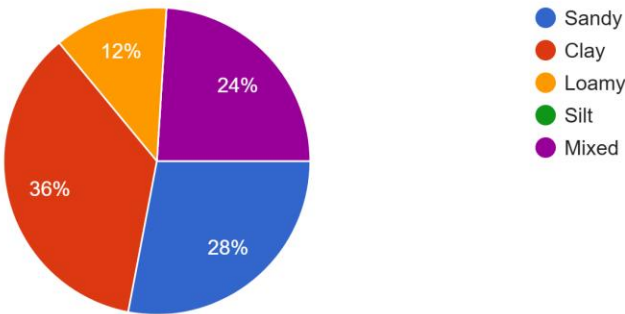


Fig. 3: Type of Soil

Which fertilizers do you mostly use?

25 responses

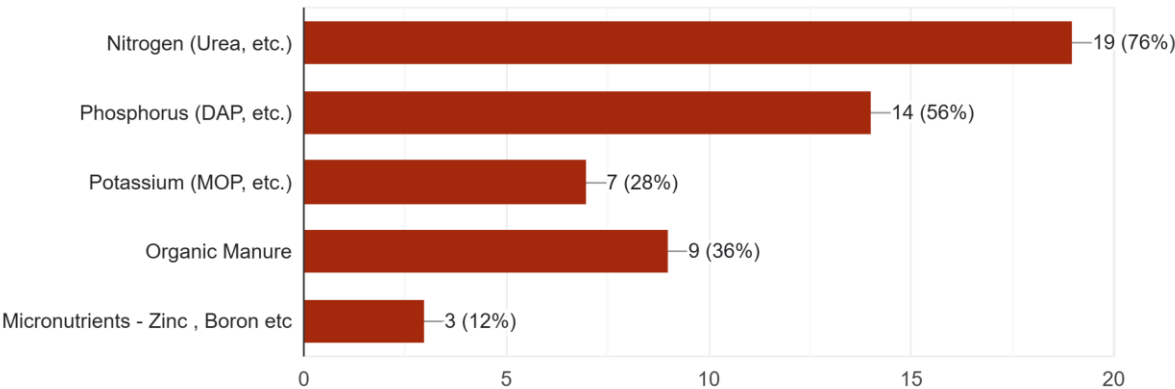


Fig. 4: Type of Fertilizers used

How do you apply fertilizers?

25 responses

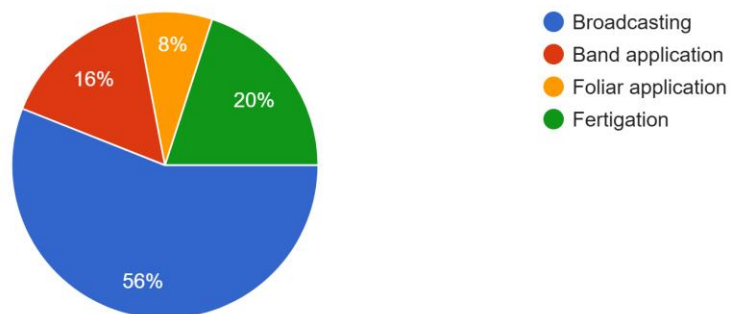


Fig. 5: Type of Fertilizer application

Where do you get information about fertilizers?

25 responses

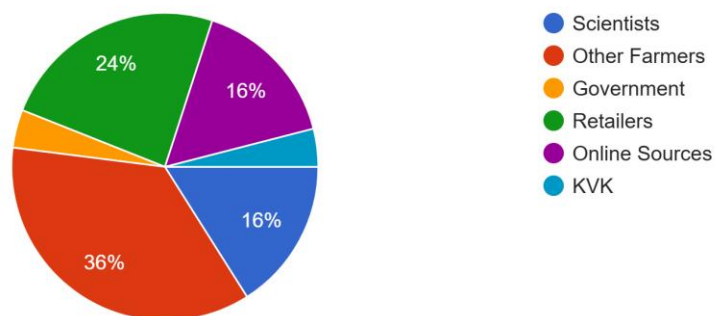


Fig. 6: Mode of information

6. OPERATING ENVIRONMENT

Hardware Requirements:

Client Devices:

The FERTIGuide application is designed to be compatible with a wide range of client devices, including smartphones, tablets, and basic desktop systems. This compatibility is essential to ensure accessibility for a diverse user demographic, particularly in rural areas where farmers may have varying levels of access to technology.

- **Smartphones:** Given the increasing penetration of mobile devices in agricultural communities, the application will be optimized for both Android and iOS platforms. This allows farmers to access real-time recommendations and input data directly from their mobile devices while working in the fields.
- **Tablets:** Tablets provide a larger screen size, which can enhance user experience for those who prefer a more detailed view of data input forms and recommendations. The application will be responsive, ensuring that it adjusts seamlessly to different screen sizes.
- **Basic Desktop Systems:** Recognizing that some users may still rely on older or less powerful desktop computers, FERTIGuide will also support basic desktop systems. This ensures that all users can access the platform regardless of their hardware capabilities.

By ensuring compatibility across these various devices, FERTIGuide aims to promote widespread adoption among farmers who may not have access to the latest technology.

Server Requirements:

To support the functionality and scalability of FERTIGuide, a robust cloud-based infrastructure is essential. This setup ensures high availability and reliability, accommodating fluctuating user demands as farmers interact with the system during peak agricultural seasons.

- **Cloud-Based Infrastructure:** Utilizing cloud services such as Amazon Web Services (AWS) or Microsoft Azure allows for dynamic resource allocation. This means that as user traffic increases—such as during planting or harvesting

seasons—the system can automatically scale to handle the load without degradation in performance.

- **High Availability:** The cloud infrastructure will implement redundancy and failover mechanisms to ensure that the application remains accessible at all times. This is crucial for farmers who rely on timely recommendations for fertilizer application based on real-time data.
- **Data Storage and Processing:** The server will host databases that store user inputs, soil health data, weather information, and fertilizer recommendations. Cloud storage solutions provide secure and scalable options for managing this data efficiently.
- **API Integration:** The server will facilitate seamless integration with external APIs for weather updates and soil health databases. This integration is vital for providing accurate and timely recommendations based on current environmental conditions.

Software Requirements

Client Software:

The FERTIGuide application will be designed to be accessible through standard web browsers and will also feature dedicated mobile applications for both Android and iOS platforms. This dual approach ensures that users can access the application from various devices, enhancing usability and convenience.

Web Application:

The web application will be developed using responsive design principles, allowing it to function effectively on desktops, laptops, tablets, and smartphones. This flexibility is crucial as it caters to a wide range of users, including those who may only have access to older or less powerful computers. The web interface will include intuitive navigation, ensuring that even users with minimal technical expertise can easily input data and interpret recommendations.

Mobile Applications:

The mobile applications will be optimized for both Android and iOS operating systems, providing farmers with on-the-go access to fertilizer recommendations. Features such as push notifications for timely alerts about weather changes or reminders for fertilizer application will enhance user engagement.

The mobile app will also support offline functionality, allowing users to access previously downloaded recommendations even in areas with poor internet connectivity.

Server Software:

To support the robust functionality of FERTIGuide, the server software will be deployed on scalable cloud platforms such as Amazon Web Services (AWS) or Microsoft Azure. This choice of infrastructure is essential for several reasons:

- **Scalability:** Cloud platforms provide the ability to scale resources up or down based on user demand. During peak agricultural seasons, when many farmers are seeking recommendations simultaneously, the system can automatically allocate additional resources to maintain performance without downtime.
- **High Availability:** Utilizing cloud infrastructure ensures high availability of the application. Redundancy features inherent in cloud services mean that if one server goes down, another can take over without affecting user access. This is particularly important for farmers who rely on timely information for fertilizer application.
- **Data Management:** The server software will facilitate efficient data handling, including storage and processing of user inputs, soil health data, weather information, and fertilizer recommendations. Cloud databases can manage large volumes of data securely while providing quick access for real-time processing.
- **API Integration:** The server will support integration with external APIs for real-time weather updates and soil health data retrieval. This capability is crucial for providing accurate recommendations based on current environmental conditions. For instance, integrating a reliable weather API allows the system to adjust fertilizer suggestions based on impending rainfall or temperature changes that may affect nutrient availability.

7. USER CLASSES AND CHARACTERISTICS

Farmers:

Farmers are the primary users of the FERTIGuide system and represent a diverse group with varying levels of technological access and literacy. Many farmers may have limited access to advanced technology, making it essential for the application to feature simple, intuitive interfaces that are easy to navigate.

- **Technological Familiarity:** Most farmers may not be familiar with complex software applications and thus require a user-friendly design. The interface should be straightforward, allowing farmers to input soil data, select crops, and receive fertilizer recommendations without needing extensive training.
- **Multi-Language Support:** Given the linguistic diversity in agricultural communities, the application must support multiple languages. This ensures that farmers from different regions can understand the information and instructions provided by the system.
- **Offline Access Options:** Many farmers operate in areas with unreliable internet connectivity. To address this, FERTIGuide will incorporate offline access features, such as SMS or USSD services, allowing users to receive recommendations and updates even without internet access.
- **Key Needs:** Farmers primarily seek actionable insights tailored to their specific agricultural conditions. They need clear guidance on fertilizer application based on local soil health and crop requirements to enhance productivity sustainably.

Agricultural Experts:

Agricultural experts serve as secondary users who validate the recommendations generated by FERTIGuide. These individuals often include extension officers or consultants with specialized knowledge in agronomy, soil science, and crop management.

- **Role in Validation:** Experts review the system-generated recommendations to ensure they align with best practices in agriculture. They provide additional insights based on their experience and may suggest adjustments based on specific local conditions or recent research findings.
- **Data Analytics Tools:** Agricultural experts require access to detailed data analytics tools that allow them to track the effectiveness of recommendations

across different regions. They need to analyze trends in crop yield improvements and assess how well the recommendations are being implemented by farmers.

- **Key Characteristics:** These users possess a high level of expertise in agriculture and often work closely with multiple farmers, providing essential support and guidance. Their role is crucial for ensuring that the recommendations made by FERTIGuide are not only accurate but also practical for real-world application.

Developers:

Developers are responsible for maintaining and enhancing the FERTIGuide system. Their expertise is vital for ensuring that the application remains functional, secure, and scalable as user demands evolve.

- **Software Development Skills:** Developers should be skilled in various programming languages and frameworks relevant to web and mobile app development. They will focus on creating a seamless user experience across all platforms while integrating machine learning models that power the recommendation engine.
- **System Maintenance:** Regular updates are necessary to fix bugs, improve performance, and introduce new features based on user feedback. Developers will monitor system performance metrics to ensure optimal operation under varying loads.
- **Key Responsibilities:** They also ensure compliance with data security standards to protect user information and maintain privacy. Their role is critical in adapting the system to changing agricultural practices or regulatory requirements.

Government Officials:

Government officials utilize FERTIGuide primarily for monitoring purposes. Their focus is on ensuring that the recommendations provided by the system align with agricultural policies and contribute positively to regional agricultural productivity.

- **Monitoring System Usage:** Officials track how effectively farmers are utilizing FERTIGuide's recommendations. This involves analyzing aggregated data to assess whether government-backed fertilizer programs or subsidies are being optimized through the platform.

8. PRODUCT FUNCTIONS

The FERTIGuide system incorporates several key product functions designed to assist farmers in optimizing fertilizer use based on accurate soil data and crop requirements. One of the critical components of this system is the soil data collection function, which enables farmers to input and manage essential soil health information. This functionality encompasses both manual and automated processes to ensure comprehensive and accurate data acquisition.

Farmers can manually enter soil test results, including parameters such as pH levels, nitrogen (N), phosphorus (P), potassium (K), moisture content, and micronutrient levels. The user interface is designed to be intuitive, guiding users through the data entry process with clear prompts and examples. To enhance accuracy and efficiency, FERTIGuide will also integrate with soil testing devices via Bluetooth or USB connections, allowing for automatic uploading of soil test results directly into the system. This integration minimizes human error and streamlines the data collection process. Once the data is collected, the system validates it to ensure completeness and accuracy, including checks for logical consistency (e.g., ensuring that pH values are within a realistic range) and completeness (e.g., confirming that all required parameters have been entered). Validated data is crucial for generating reliable fertilizer recommendations.

The **crop recommendation engine** serves as the core functionality of FERTIGuide, utilizing advanced algorithms and machine learning models to provide tailored fertilizer recommendations based on real-time inputs. This engine leverages machine learning techniques to analyze historical data on soil health, crop types, and weather conditions. By identifying patterns and correlations, the system can generate precise recommendations that optimize nutrient application for specific crops. Recognizing that soil types and weather conditions vary significantly across regions, the recommendation engine incorporates geographic data to adjust suggestions accordingly. For example, a recommendation for rice cultivation in a humid region may differ from that in a drier climate due to differences in nutrient availability and crop water requirements.

Real-time recommendations are a vital aspect of this engine; by integrating weather data through external APIs, the system can dynamically adjust recommendations based on current environmental conditions. For instance, if rain is forecasted shortly after fertilizer application, the system may suggest delaying application to prevent runoff. The recommendations will include specific dosage amounts for each nutrient as well as

guidance on application frequency, ensuring that farmers can apply fertilizers in optimal quantities and at appropriate times to maximize crop yield while minimizing waste.

The **user feedback mechanism** is designed to facilitate continuous improvement of the FERTIGuide system by collecting insights from users regarding their experiences with the recommendations provided. Farmers will have the option to submit feedback on the effectiveness of the fertilizer recommendations they implemented. This feedback can include metrics such as crop yield improvements, any observed deficiencies or excesses in nutrient application, and overall satisfaction with the recommendations. To enhance user support, this mechanism allows farmers to connect with agricultural experts for personalized advice. If a farmer encounters issues or has questions about specific recommendations, they can easily reach out for expert guidance through the platform.

Additionally, the system will track user feedback over time to assess the long-term effectiveness of its recommendations. By analyzing this data, FERTIGuide can refine its algorithms and improve future recommendations based on real-world outcomes. This iterative process ensures that the system evolves in response to user needs and changing agricultural practices.

9. ASSUMPTIONS AND CONSTRAINTS

Assumptions:

1. Farmers have access to soil testing equipment or nearby facilities.
2. Users will have reliable internet connections in urban areas.
3. Farmers possess a basic level of familiarity with mobile devices and web applications.
4. Training resources or support systems will be available to help farmers understand how to use the FERTIGuide application.
5. Agricultural experts will be available to validate recommendations and provide additional guidance.
6. Agricultural practices vary significantly across different regions, necessitating tailored recommendations.
7. Farmers will be willing to implement the fertilizer recommendations provided by FERTIGuide.
8. Farmers are committed to sustainable agricultural practices and consider environmental impacts when applying fertilizers.
9. Government policies will support the use of technologies like FERTIGuide for improving soil health and crop productivity.

Constraints:

1. Limited access to technology among some farmers.
2. Variability in internet connectivity in rural regions.
3. Dependence on traditional farming practices may hinder technology adoption.
4. High costs associated with soil testing services may deter usage.
5. Resistance to change from established farming methods to data-driven approaches.
6. Inconsistent availability of agricultural extension services for training and support.
7. Potential language barriers in user interfaces may limit accessibility for non-native speakers.
8. Limited financial resources for farmers to invest in necessary technology or services.

10. BUSINESS CONSTRAINTS

Budget limitations for marketing the application:

One of the significant constraints facing the FERTIGuide project is budget limitations for marketing the application. Effective marketing is crucial for ensuring that the target audience—primarily farmers and agricultural experts—becomes aware of the application and its benefits. However, limited financial resources can restrict the scope and reach of marketing efforts. This constraint may lead to reliance on low-cost marketing strategies, such as social media campaigns or community outreach programs, which may not achieve the desired level of visibility. Additionally, without adequate funding, there may be challenges in developing promotional materials, conducting workshops, or engaging in partnerships with agricultural organizations that could facilitate broader adoption of the application. Consequently, budget limitations could hinder the overall impact of FERTIGuide, affecting its ability to reach and assist a larger number of farmers.

Regulatory compliance with agricultural policies:

Another critical constraint involves regulatory compliance with agricultural policies. The FERTIGuide application must adhere to various local and national regulations governing agricultural practices, including those related to fertilizer use and environmental protection. Compliance with these regulations is essential to ensure that the recommendations provided by the application align with legal standards and best practices in sustainable agriculture. This requirement can complicate the development process, as it necessitates thorough research into existing policies and potential changes in regulations over time. Moreover, any updates or modifications to the application may require additional validation to ensure ongoing compliance, which can consume valuable resources and time. Failure to comply with regulatory requirements could result in legal repercussions or damage to the application's credibility among users.

Need for collaboration with government agencies for data integration:

The success of FERTIGuide also hinges on effective collaboration with government agencies for data integration. Accessing reliable soil health data, weather information, and other relevant agricultural datasets is vital for generating accurate fertilizer recommendations. Collaborating with government agencies can facilitate this data integration by providing access to national databases and ensuring that the

information used by FERTIGuide is up-to-date and comprehensive. However, establishing these collaborations can be challenging due to bureaucratic processes or differing priorities between agencies and the project team. Furthermore, ongoing collaboration may be necessary to maintain data accuracy as new research emerges or as agricultural policies evolve. The need for such partnerships underscores the importance of building strong relationships with governmental bodies to enhance the effectiveness of FERTIGuide while navigating potential obstacles related to data sharing and integration.

RELATIONAL MODEL DIAGRAM

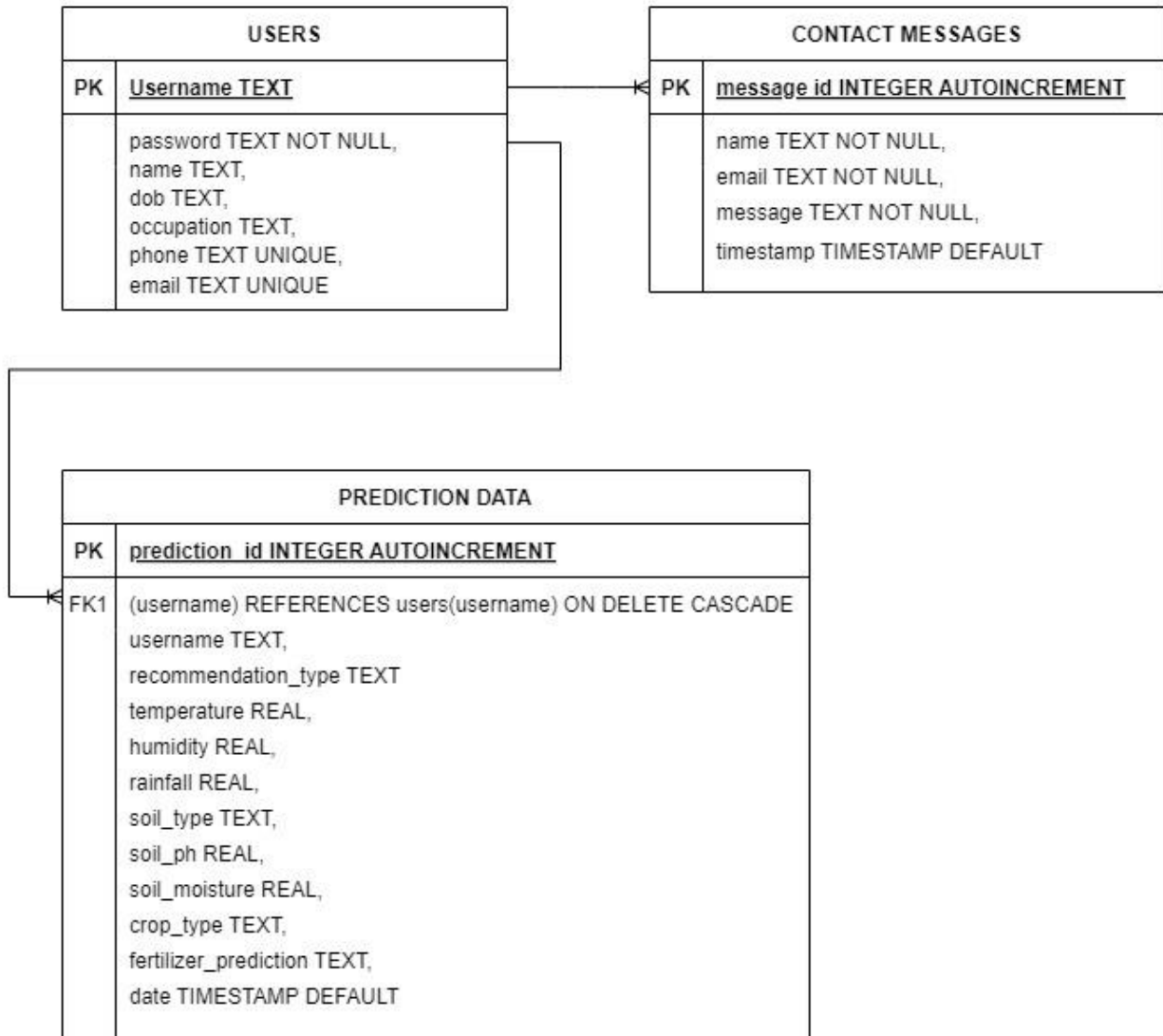


Fig. 7: ER Model of FERTIGuide

This image represents an **Entity-Relationship Diagram (ERD)** for a system involving users, contact messages, and prediction data.

USERS Table

This table stores user-related information.

- **Primary Key (PK):** username (TEXT)
- **Attributes:**

- password (TEXT NOT NULL): User's password.
- name (TEXT): User's name.
- dob (TEXT): Date of birth.
- occupation (TEXT): User's occupation.
- phone (TEXT UNIQUE): User's phone number (unique constraint).
- email (TEXT UNIQUE): User's email address (unique constraint).

CONTACT MESSAGES Table

This table stores messages sent by users or external contacts.

- **Primary Key (PK):** message_id (INTEGER AUTOINCREMENT)
- **Attributes:**
 - name (TEXT NOT NULL): Name of the person sending the message.
 - email (TEXT NOT NULL): Email address of the sender.
 - message (TEXT NOT NULL): The content of the message.
 - timestamp (TIMESTAMP DEFAULT): Automatically stores the date and time the message was sent.

PREDICTION DATA Table

This table records prediction-related information for users.

- **Primary Key (PK):** prediction_id (INTEGER AUTOINCREMENT)
- **Foreign Key (FK1):** username references the username column in the **USERS** table. Includes ON DELETE CASCADE to delete associated prediction data when the user is deleted.
- **Attributes:**
 - username (TEXT): Links to a specific user.
 - recommendation_type (TEXT): Type of recommendation provided.
 - temperature (REAL): Temperature data used in predictions.
 - humidity (REAL): Humidity data.
 - rainfall (REAL): Rainfall data.
 - soil_type (TEXT): Type of soil.
 - soil_ph (REAL): Soil pH value.
 - soil_moisture (REAL): Soil moisture content.

- crop (TEXT): Suggested crop based on predictions.
- fertilizer_prediction (TEXT): Fertilizer recommendations.
- date (TIMESTAMP DEFAULT): Date and time of the prediction.

RELATIONSHIP

1. USERS to PREDICTION DATA:

- a. One-to-Many relationship (username in **PREDICTION DATA** references username in **USERS**). A single user can have multiple predictions.

2. USERS to CONTACT MESSAGES:

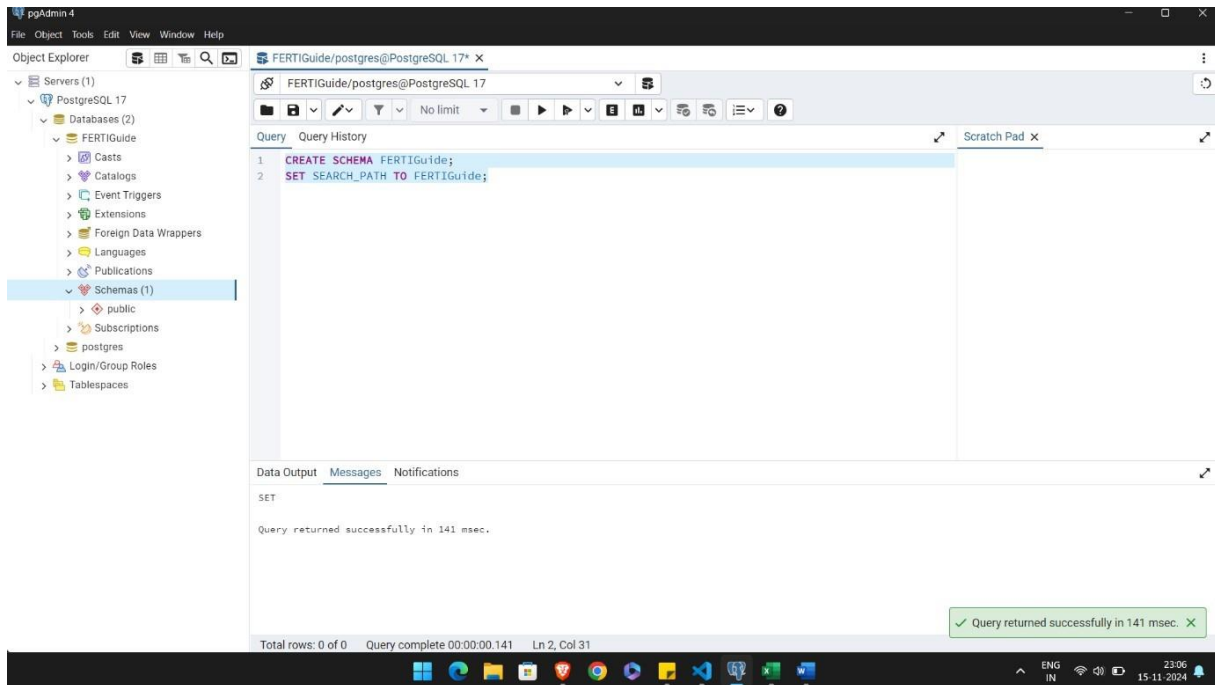
- a. Not directly related in this diagram, but the **CONTACT MESSAGES** table is standalone for communication purposes.

DDL SCRIPTS AND SQL QUERIES

DDL Script:

```
CREATE SCHEMA FERTIGuide;
```

```
SET SEARCH_PATH TO FERTIGuide;
```



```
SET SEARCH_PATH TO FERTIGuide;
```

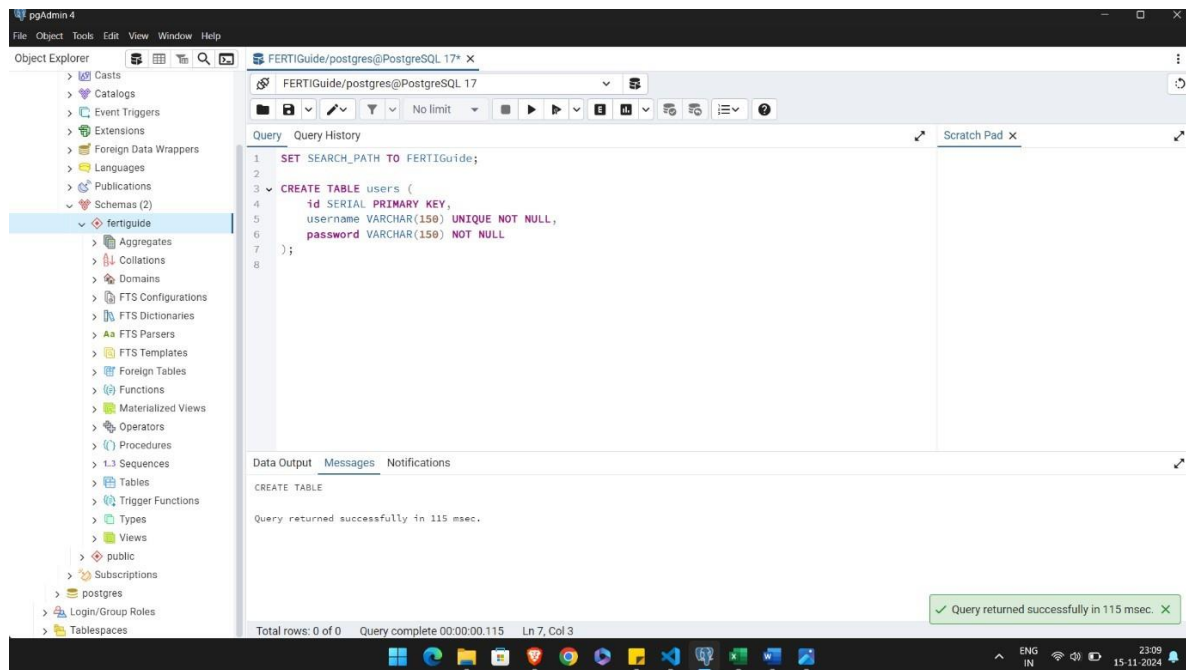
```
CREATE TABLE users (
```

```
    id SERIAL PRIMARY KEY,
```

```
    username VARCHAR(150) UNIQUE NOT NULL,
```

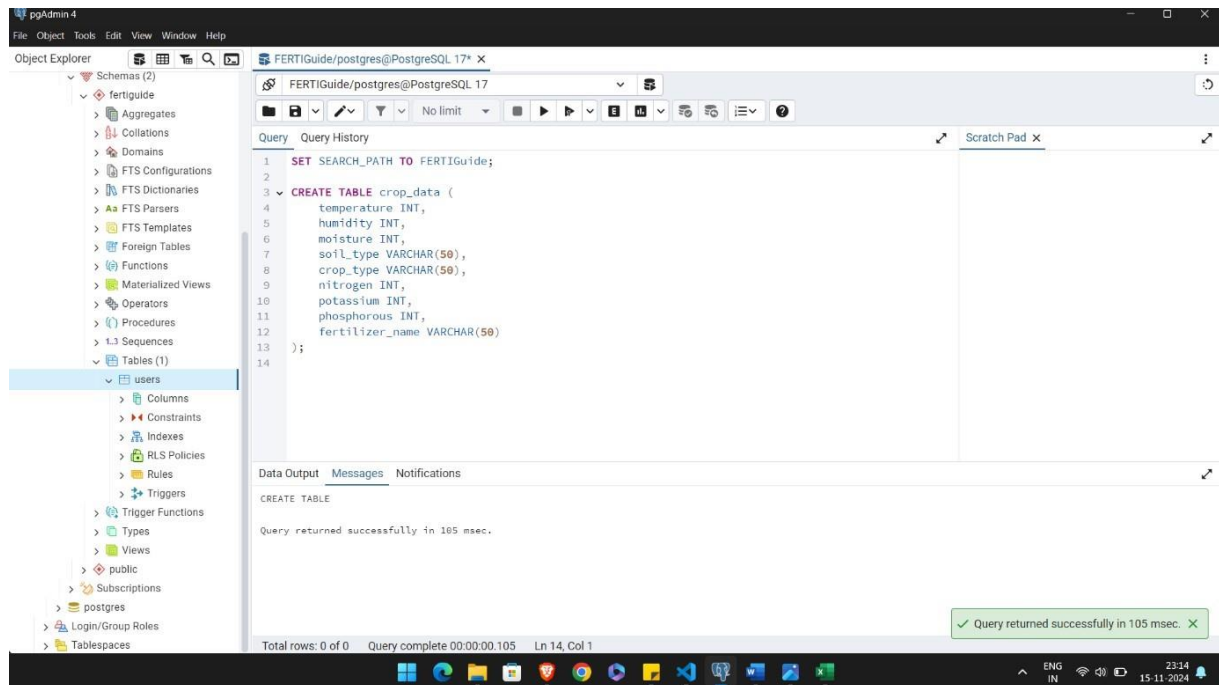
```
    password VARCHAR(150) NOT NULL
```

```
);
```



SET SEARCH_PATH TO FERTIGuide;

```
CREATE TABLE crop_data  
( temperature INT,  
  humidity INT,  
  
  moisture INT,  
  
  soil_type VARCHAR(50),  
  crop_type VARCHAR(50),  
  nitrogen INT,  
  
  potassium INT,  
  phosphorous INT,  
  
  fertilizer_name VARCHAR(50)  
);
```



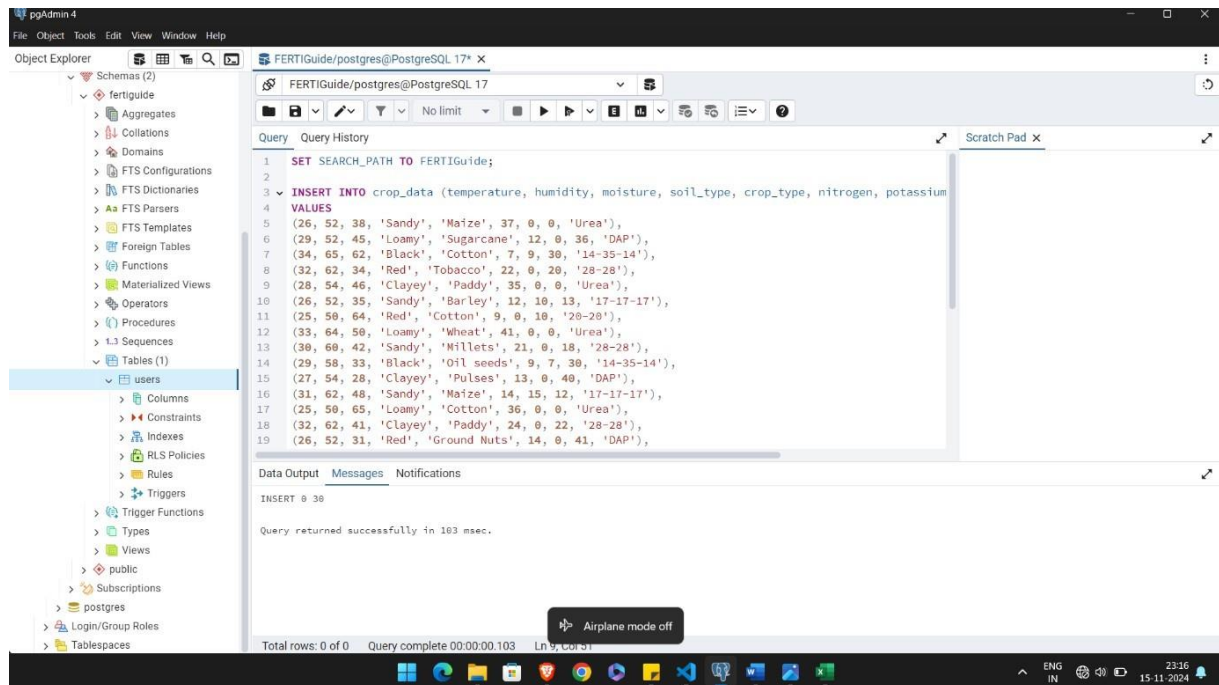
SET SEARCH_PATH TO FERTIGuide;

INSERT INTO crop_data (temperature, humidity, moisture, soil_type, crop_type, nitrogen, potassium, phosphorous, fertilizer_name)

VALUES

(26, 52, 38, 'Sandy', 'Maize', 37, 0, 0, 'Urea'),
 (29, 52, 45, 'Loamy', 'Sugarcane', 12, 0, 36, 'DAP'),
 (34, 65, 62, 'Black', 'Cotton', 7, 9, 30, '14-35-14'),
 (32, 62, 34, 'Red', 'Tobacco', 22, 0, 20, '28-28'),
 (28, 54, 46, 'Clayey', 'Paddy', 35, 0, 0, 'Urea'),
 (26, 52, 35, 'Sandy', 'Barley', 12, 10, 13, '17-17-17'),
 (25, 50, 64, 'Red', 'Cotton', 9, 0, 10, '20-20'),
 (33, 64, 50, 'Loamy', 'Wheat', 41, 0, 0, 'Urea'),
 (30, 60, 42, 'Sandy', 'Millets', 21, 0, 18, '28-28'),
 (29, 58, 33, 'Black', 'Oil seeds', 9, 7, 30, '14-35-14'),
 (27, 54, 28, 'Clayey', 'Pulses', 13, 0, 40, 'DAP'),

(31, 62, 48, 'Sandy', 'Maize', 14, 15, 12, '17-17-17'),
(25, 50, 65, 'Loamy', 'Cotton', 36, 0, 0, 'Urea'),
(32, 62, 41, 'Clayey', 'Paddy', 24, 0, 22, '28-28'),
(26, 52, 31, 'Red', 'Ground Nuts', 14, 0, 41, 'DAP'),
(31, 62, 49, 'Black', 'Sugarcane', 10, 13, 14, '17-17-17'),
(33, 64, 34, 'Clayey', 'Pulses', 38, 0, 0, 'Urea'),
(25, 50, 39, 'Sandy', 'Barley', 21, 0, 19, '28-28'),
(28, 54, 65, 'Black', 'Cotton', 39, 0, 0, 'Urea'),
(29, 58, 52, 'Loamy', 'Wheat', 13, 0, 36, 'DAP'),
(30, 60, 44, 'Sandy', 'Millets', 10, 0, 9, '20-20'),
(34, 65, 53, 'Loamy', 'Sugarcane', 12, 14, 12, '17-17-17'),
(35, 68, 33, 'Red', 'Tobacco', 11, 0, 37, 'DAP'),
(28, 54, 37, 'Black', 'Millets', 36, 0, 0, 'Urea'),
(33, 64, 39, 'Clayey', 'Paddy', 13, 0, 10, '20-20'),
(26, 52, 44, 'Sandy', 'Maize', 23, 0, 20, '28-28'),
(30, 60, 63, 'Red', 'Cotton', 9, 9, 29, '14-35-14'),
(32, 62, 30, 'Loamy', 'Sugarcane', 38, 0, 0, 'Urea'),
(37, 70, 32, 'Black', 'Oil seeds', 12, 0, 39, 'DAP'),
(26, 52, 36, 'Clayey', 'Pulses', 14, 0, 13, '20-20');



SET SEARCH_PATH TO FERTIGuide;

SELECT temperature,
humidity,

moisture,
soil_type,
crop_type,
nitrogen,
potassium,
phosphorous,

fertilizer_name
FROM crop_data;

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer

- > Collations
- > Domains
- > FTS Configurations
- > FTS Dictionaries
- > FTS Parsers
- > FTS Templates
- > Foreign Tables
- > Functions
- > Materialized Views
- > Operators
- > Procedures
- > Sequences
- > Tables (2)
 - > crop_data
 - > Columns (9)
 - temperature
 - humidity
 - moisture
 - soil_type
 - crop_type
 - nitrogen
 - potassium
 - phosphorous
 - fertilizer_name
 - > Constraints
 - > Indexes
 - > RLS Policies
 - > Rules
 - > Triggers
 - > users
 - > Trigger Functions

FERTIGuide/postgres@PostgreSQL 17*

Query Query History

```

1 SET SEARCH_PATH TO FERTIGuide;
2
3 SELECT temperature,
4 humidity,
5 moisture,
6 soil_type,
7 crop_type,
8 nitrogen,
9 potassium,
10 phosphorous,
11 fertilizer_name
12 FROM crop_data;

```

Data Output Messages Notifications

	temperature integer	humidity integer	moisture integer	soil_type character varying (50)	crop_type character varying (50)	nitrogen integer	potassium integer	phosphorous integer	fertilizer_name character varying (50)
1	26	52	38	Sandy	Maize	37	0	0	Urea
2	29	52	45	Loamy	Sugarcane	12	0	36	DAP
3	34	65	62	Black	Cotton	7	9	30	14-35-14
4	32	62	34	Red	Tobacco	22	0	20	28-28
5	28	54	46	Clayey	Paddy	35	0	0	Urea
6	26	52	35	Sandy	Barley	12	10	13	17-17-17
7	25	50	64	Red	Cotton	9	0	10	20-20
8	33	64	50	Loamy	Wheat	41	0	0	Urea
9	30	60	42	Sandy	Millets	21	0	18	28-28
10	29	58	33	Black	Oil seeds	9	7	30	14-35-14

Total rows: 30 of 30 Query complete 00:00:00.162 Ln 12, Col 1

SIMPLE QUERIES:

SET SEARCH_PATH TO FERTIGuide;

**SELECT * FROM crop_data
WHERE crop_type = 'Maize';**

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer

- > Collations
- > Domains
- > FTS Configurations
- > FTS Dictionaries
- > FTS Parsers
- > FTS Templates
- > Foreign Tables
- > Functions
- > Materialized Views
- > Operators
- > Procedures
- > Sequences
- > Tables (2)
 - > crop_data
 - > Columns (9)
 - temperature
 - humidity
 - moisture
 - soil_type
 - crop_type
 - nitrogen
 - potassium
 - phosphorous
 - fertilizer_name
 - > Constraints
 - > Indexes
 - > RLS Policies
 - > Rules
 - > Triggers
 - > users
 - > Trigger Functions

FERTIGuide/postgres@PostgreSQL 17*

Query Query History

```

1 SET SEARCH_PATH TO FERTIGuide;
2
3 SELECT * FROM crop_data
4 WHERE crop_type = 'Maize';

```

Data Output Messages Notifications

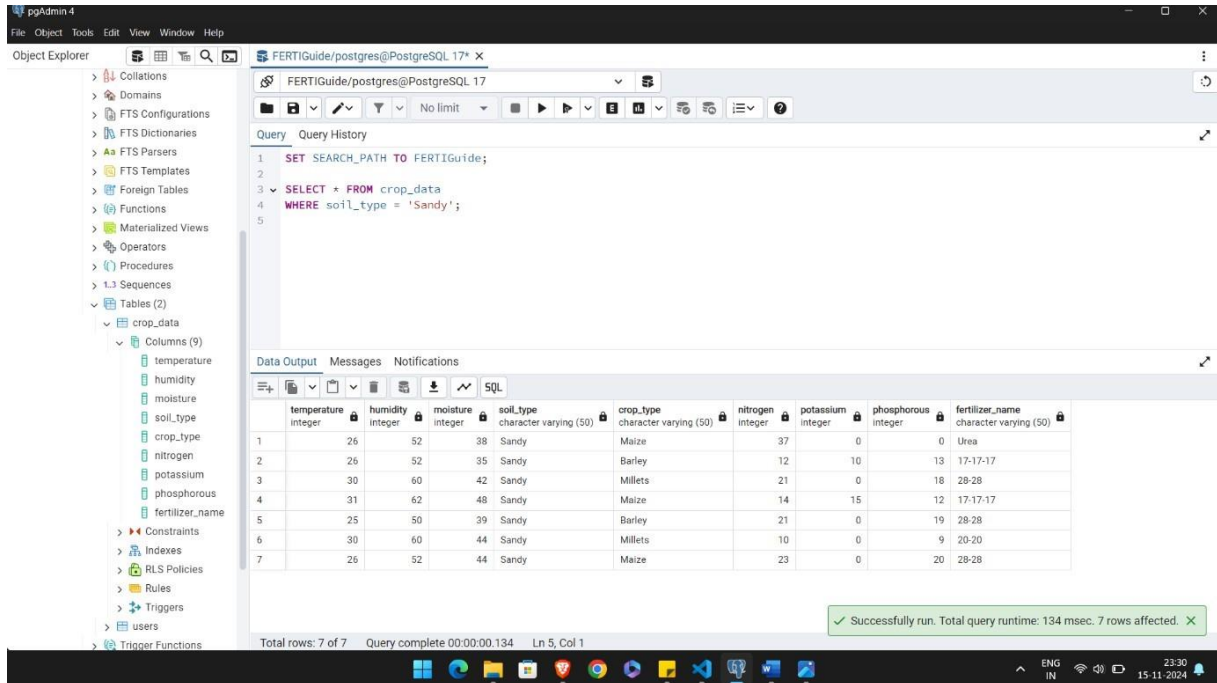
	temperature integer	humidity integer	moisture integer	soil_type character varying (50)	crop_type character varying (50)	nitrogen integer	potassium integer	phosphorous integer	fertilizer_name character varying (50)
1	26	52	38	Sandy	Maize	37	0	0	Urea
2	31	62	48	Sandy	Maize	14	15	12	17-17-17
3	26	52	44	Sandy	Maize	23	0	20	28-28

Successfully run. Total query runtime: 155 msec. 3 rows affected.

Total rows: 3 of 3 Query complete 00:00:00.155 Ln 4, Col 27

SET SEARCH_PATH TO FERTIGuide;

**SELECT * FROM crop_data
WHERE soil_type = 'Sandy';**



pgAdmin 4

Object Explorer

- Collations
- Domains
- FTS Configurations
- FTS Dictionaries
- FTS Parsers
- FTS Templates
- Foreign Tables
- Functions
- Materialized Views
- Operators
- Procedures
- Sequences
- Tables (2)
 - crop_data
 - Columns (9)
 - temperature
 - humidity
 - moisture
 - soil_type
 - crop_type
 - nitrogen
 - potassium
 - phosphorous
 - fertilizer_name
- Constraints
- Indexes
- RLS Policies
- Rules
- Triggers
- users
- Trigger Functions

Query

```
1 SET SEARCH_PATH TO FERTIGuide;  
2  
3 SELECT * FROM crop_data  
4 WHERE soil_type = 'Sandy';  
5
```

Data Output

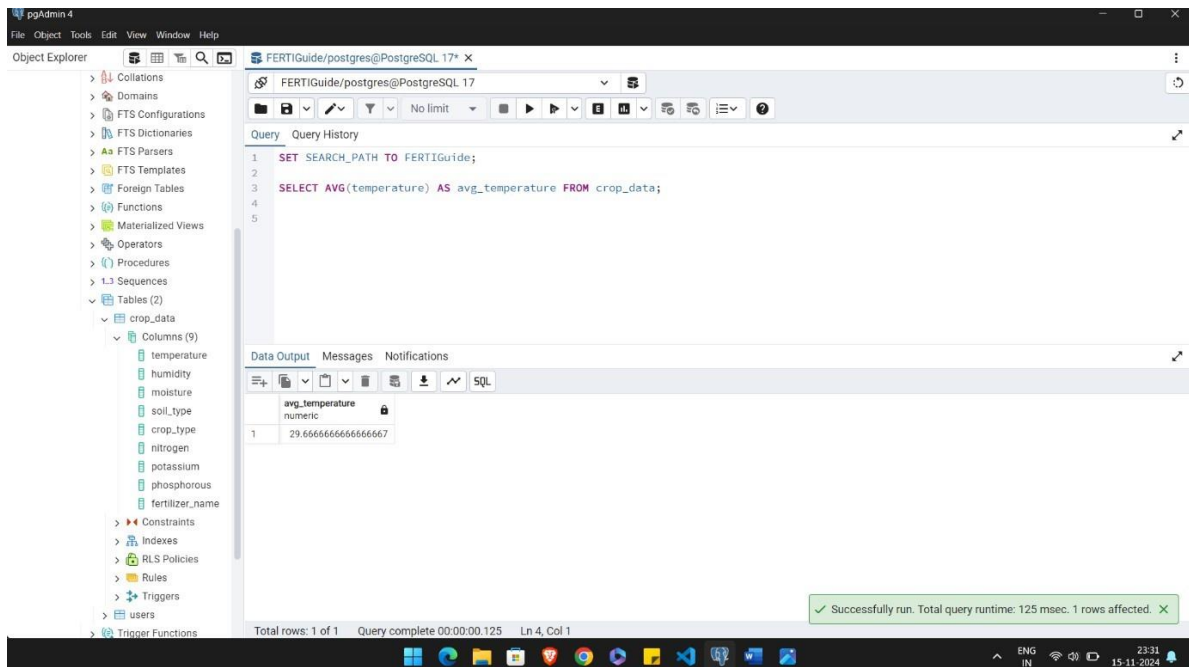
	temperature integer	humidity integer	moisture integer	soil_type character varying (50)	crop_type character varying (50)	nitrogen integer	potassium integer	phosphorous integer	fertilizer_name character varying (50)
1	26	52	38	Sandy	Maize	37	0	0	Urea
2	26	52	35	Sandy	Barley	12	10	13	17-17-17
3	30	60	42	Sandy	Millets	21	0	18	28-28
4	31	62	48	Sandy	Maize	14	15	12	17-17-17
5	25	50	39	Sandy	Barley	21	0	19	28-28
6	30	60	44	Sandy	Millets	10	0	9	20-20
7	26	52	44	Sandy	Maize	23	0	20	28-28

Successfully run. Total query runtime: 134 msec. 7 rows affected.

Total rows: 7 of 7 Query complete 00:00:00.134 Ln 5, Col 1

SET SEARCH_PATH TO FERTIGuide;

SELECT AVG(temperature) AS avg_temperature FROM crop_data;



pgAdmin 4

Object Explorer

- Collations
- Domains
- FTS Configurations
- FTS Dictionaries
- FTS Parsers
- FTS Templates
- Foreign Tables
- Functions
- Materialized Views
- Operators
- Procedures
- Sequences
- Tables (2)
 - crop_data
 - Columns (9)
 - temperature
 - humidity
 - moisture
 - soil_type
 - crop_type
 - nitrogen
 - potassium
 - phosphorous
 - fertilizer_name
- Constraints
- Indexes
- RLS Policies
- Rules
- Triggers
- users
- Trigger Functions

Query

```
1 SET SEARCH_PATH TO FERTIGuide;  
2  
3 SELECT AVG(temperature) AS avg_temperature FROM crop_data;  
4  
5
```

Data Output

	avg_temperature numeric
1	29.666666666666667

Successfully run. Total query runtime: 125 msec. 1 rows affected.

Total rows: 1 of 1 Query complete 00:00:00.125 Ln 4, Col 1

SET SEARCH_PATH TO FERTIGuide;

SELECT SUM(nitrogen) AS total_nitrogen FROM crop_data;

The screenshot shows the pgAdmin 4 interface. The left pane displays the database structure, including the 'crop_data' table with columns: temperature, humidity, moisture, soil_type, crop_type, nitrogen, potassium, phosphorous, and fertilizer_name. The central query editor contains the following SQL code:

```
1 SET SEARCH_PATH TO FERTIGuide;
2
3 SELECT SUM(nitrogen) AS total_nitrogen FROM crop_data;
4
5
```

The 'Data Output' pane shows the result of the query:

total_nitrogen
605

A status message at the bottom indicates: 'Successfully run. Total query runtime: 116 msec. 1 rows affected.'

SET SEARCH_PATH TO FERTIGuide;

SELECT MAX(moisture) AS max_moisture FROM crop_data;

The screenshot shows the pgAdmin 4 interface. The left pane displays the database structure, including the 'crop_data' table with columns: temperature, humidity, moisture, soil_type, crop_type, nitrogen, potassium, phosphorous, and fertilizer_name. The central query editor contains the following SQL code:

```
1 SET SEARCH_PATH TO FERTIGuide;
2
3 SELECT MAX(moisture) AS max_moisture FROM crop_data;
4
5
6
```

The 'Data Output' pane shows the result of the query:

max_moisture
65

A status message at the bottom indicates: 'Successfully run. Total query runtime: 131 msec. 1 rows affected.'

COMPLEX QUERIES:

SET SEARCH_PATH TO FERTIGuide;

SELECT crop_type, fertilizer_name
FROM crop_data

WHERE nitrogen > 20;

The screenshot shows the pgAdmin 4 interface. The left pane displays the Object Explorer with the 'crop_data' table selected. The main pane shows the SQL editor with the following query:

```
1 SET SEARCH_PATH TO FERTIGuide;
2
3 SELECT crop_type, fertilizer_name
4 FROM crop_data
5 WHERE nitrogen > 20;
6
7
8
9
```

The 'Data Output' pane shows the results of the query:

crop_type	fertilizer_name
Maize	Urea
Tobacco	28-28
Paddy	Urea
Wheat	Urea
Millet	28-28
Cotton	Urea
Paddy	28-28
Pulses	Urea
Barley	28-28
Cotton	Urea

A status bar at the bottom indicates: 'Successfully run. Total query runtime: 294 msec. 13 rows affected.'

SET SEARCH_PATH TO FERTIGuide;

SELECT crop_type, soil_type, nitrogen
FROM crop_data

WHERE nitrogen < 20;

The screenshot shows the pgAdmin 4 interface. The left pane displays the Object Explorer with the 'crop_data' table selected. The main pane shows the SQL editor with the following query:

```
1 SET SEARCH_PATH TO FERTIGuide;
2
3 SELECT crop_type, soil_type, nitrogen
4 FROM crop_data
5 WHERE nitrogen < 20;
6
```

The 'Data Output' pane shows the results of the query:

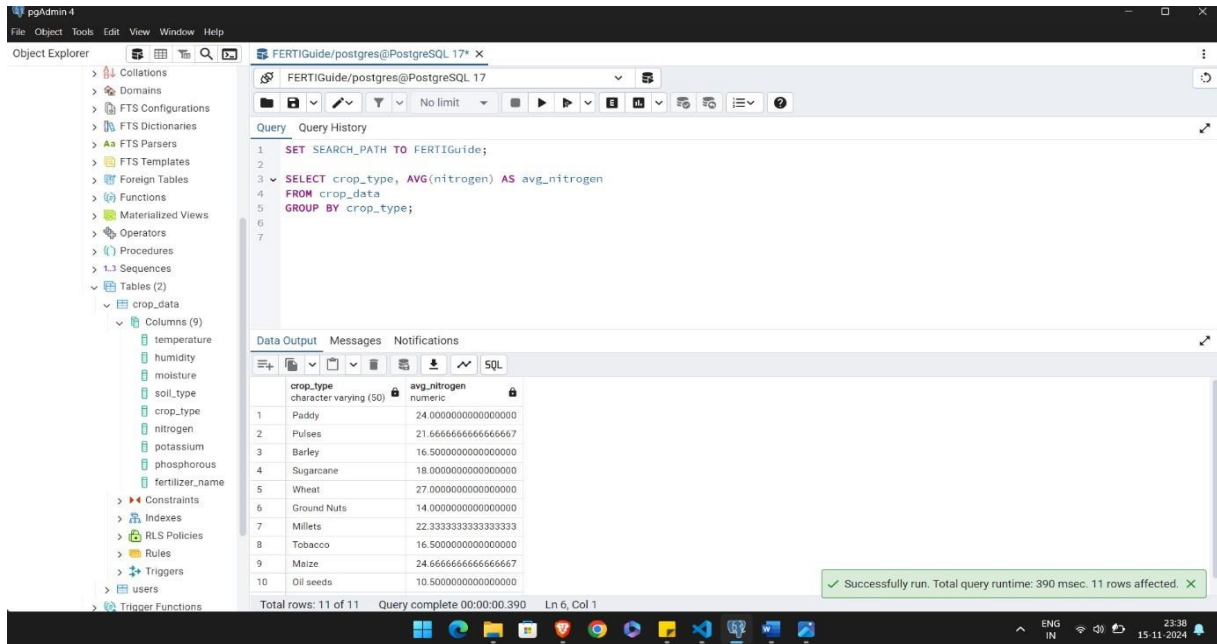
crop_type	soil_type	nitrogen
Sugarcane	Loamy	12
Cotton	Black	7
Barley	Sandy	12
Cotton	Red	9
Oil seeds	Black	9
Pulses	Clayey	13
Maize	Sandy	14
Ground Nuts	Red	14
Sugarcane	Black	10
Wheat	Loamy	13

A status bar at the bottom indicates: 'Successfully run. Total query runtime: 294 msec. 17 rows affected.'

SET SEARCH_PATH TO FERTIGuide;

**SELECT crop_type, AVG(nitrogen) AS avg_nitrogen
FROM crop_data**

GROUP BY crop_type;



The screenshot shows the pgAdmin 4 interface with the following SQL query executed:

```
1 SET SEARCH_PATH TO FERTIGuide;
2
3 SELECT crop_type, AVG(nitrogen) AS avg_nitrogen
4 FROM crop_data
5 GROUP BY crop_type;
```

The Data Output tab displays the results of the query:

crop_type	avg_nitrogen
Paddy	24.000000000000000
Pulses	21.666666666666667
Barley	16.500000000000000
Sugarcane	18.000000000000000
Wheat	27.000000000000000
Ground Nuts	14.000000000000000
Millets	22.333333333333333
Tobacco	16.500000000000000
Maize	24.666666666666667
Oil seeds	10.500000000000000

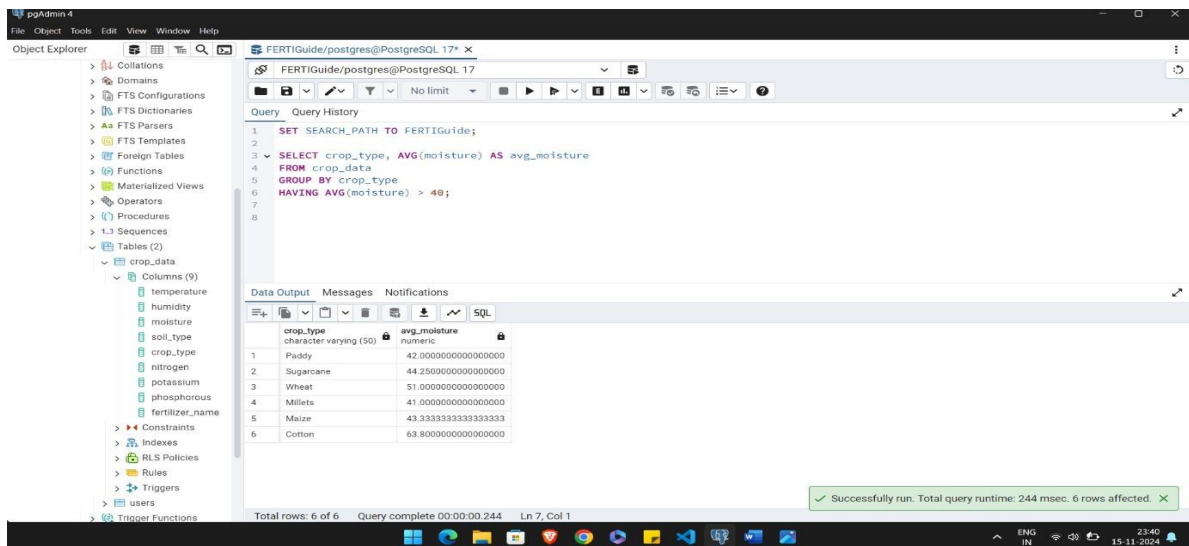
Successfully run. Total query runtime: 390 msec. 11 rows affected.

SET SEARCH_PATH TO FERTIGuide;

**SELECT crop_type, AVG(moisture) AS avg_moisture
FROM crop_data**

GROUP BY crop_type

HAVING AVG(moisture) > 40;



The screenshot shows the pgAdmin 4 interface with the following SQL query executed:

```
1 SET SEARCH_PATH TO FERTIGuide;
2
3 SELECT crop_type, AVG(moisture) AS avg_moisture
4 FROM crop_data
5 GROUP BY crop_type
6 HAVING AVG(moisture) > 40;
```

The Data Output tab displays the results of the query:

crop_type	avg_moisture
Paddy	42.000000000000000
Sugarcane	44.250000000000000
Wheat	51.000000000000000
Millets	41.000000000000000
Maize	43.333333333333333
Cotton	63.800000000000000

Successfully run. Total query runtime: 244 msec. 6 rows affected.

SET SEARCH_PATH TO FERTIGuide;

**SELECT crop_type, temperature
FROM crop_data**

ORDER BY temperature DESC;

The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer displays the database structure, including the 'crop_data' table. The main query editor contains the following SQL code:

```
1 SET SEARCH_PATH TO FERTIGuide;
2
3 SELECT crop_type, temperature
4 FROM crop_data
5 ORDER BY temperature DESC;
6
7
8
```

The 'Data Output' pane shows the results of the query, displaying two columns: 'crop_type' (character varying (50)) and 'temperature' (integer). The results are sorted by temperature in descending order.

crop_type	temperature
Oil seeds	37
Tobacco	35
Sugarcane	34
Cotton	34
Wheat	33
Pulses	33
Paddy	33
Tobacco	32
Sugarcane	32
Paddy	32

A status message at the bottom right indicates: 'Successfully run. Total query runtime: 235 msec. 30 rows affected.'

Contribution (Team-wise)

The FERTIGuide project was a collaborative effort between Deepthi Kumari Rayapureddy and Nidhi Chaudhary, with both of us contributing significantly to different aspects of the project. I focused on the technical development of the application, particularly in designing the machine learning algorithms that power the crop recommendation engine. I was responsible for integrating various data sources, including soil health metrics and weather conditions, to ensure accurate and actionable recommendations for farmers. Also designed the database management scripts necessary for storing and retrieving data efficiently within the application. Both of us worked closely together to integrate our respective components, ensuring a seamless workflow from data collection and model development to final reporting and database management.

Nidhi took charge of documentation and project management. She created the Software Requirements Specification (SRS), ensuring that all functional and non-functional requirements were clearly defined and aligned with user needs. Additionally, prepared the final project report, summarizing the development process, challenges faced, and outcomes achieved. The success of FERTIGuide was made possible through our collective effort, where we each brought our unique skills to different facets of the work, contributing to the overall goal of creating an effective fertilizer recommendation system for farmers.

Overall take away from the course

The Fundamentals of Python Programming with SQL course at DAIICT, Gandhinagar provided an enriching experience that established a strong foundation in both Python programming and SQL. Under the guidance of Kalgi Gandhi Mam, the course offered an in-depth exploration of Python and SQL, particularly focusing on their applications in data manipulation and analysis.

In the Python segment, we started with the fundamentals, covering essential concepts such as variables, data types, conditional statements, loops, and functions. We also explored significant Python libraries like pandas, numpy, and matplotlib, which are vital for data manipulation, statistical analysis, and data visualization. These libraries equipped us to handle datasets efficiently, perform calculations, and visualize data effectively, enabling us to derive meaningful insights from raw information.

On the SQL side, the course introduced us to the basics of relational databases and how to interact with them using SQL. We learned to write fundamental SQL queries for selecting, filtering, sorting, and aggregating data from databases. Additionally, we examined more advanced SQL concepts such as joins and grouping, which allowed us to merge multiple data sources and execute complex data operations. By the end of the course, we gained a solid understanding of how to structure and query databases effectively to retrieve and analyze relevant data.

Throughout the course, there was a strong emphasis on practical application through assignments and hands-on projects that helped us apply the concepts learned. The course not only provided a robust theoretical foundation but also instilled confidence in working with real-world datasets, cleaning and manipulating them using Python, and analyzing them with SQL. Overall, this course was a pivotal step in our learning journey, equipping us with essential tools for data analysis and preparing us to explore further into the field of data science. It has laid the groundwork for confidently tackling more advanced topics in data analytics, machine learning, and beyond.