SOIL-INFORMATION SYSTEM

ABSTRACT

The Soil Information System is a Python-based mini project designed to provide detailed information about various soil types and recommend suitable crops based on soil properties. The project aims to bridge the gap between agricultural knowledge and technology by offering an interactive, menu-driven software solution that mimics the functionality of a basic expert system for farmers or agricultural learners.

Built using an object-oriented programming approach, the system stores predefined data on major Indian soil types, including their nitrogen, phosphorus, potassium content, pH values, suitable crops, and regions where they are commonly found. Users can explore soil types, retrieve detailed information, and receive crop-based soil recommendations. Additionally, the system allows saving selected soil data to a local file for offline access.

This project highlights the practical application of core Python concepts such as class design, dictionaries, file handling, and user interaction. It also provides a foundational model for future integration with real-world sensor hardware to detect soil parameters dynamically. The Soil Information System offers both educational value and real-life relevance in the context of smart agriculture and precision farming.

TABLE OF CONTENTS

1: Introduction

Agriculture is a cornerstone of human civilization, and the quality of soil directly influences the success of agricultural activities. Different crops require specific soil conditions for optimal growth, including the right balance of nutrients such as nitrogen (N), phosphorus (P), potassium (K), and an appropriate pH level. Understanding and managing this soil data is crucial for farmers, agricultural students, researchers, and policymakers.

The Soil Information System is a Python-based mini project developed to simplify and digitize this knowledge. It acts as a basic decision-support tool, helping users:

Identify and learn about various types of soil found across different regions.Understand the nutrient and pH profile of each soil type.Get recommendations on which soil types are best suited for a specific crop.Export and save soil information into text files for offline reference.

This project is designed using Object-Oriented Programming (OOP) principles, providing a clear structure and reusability. The system is interactive and menu-driven, making it easy for users to navigate through the options without needing any technical background.

By integrating soil data and agricultural recommendations into a simple Python application, this project not only reinforces programming concepts like classes, methods, file handling, and user input handling but also showcases how technology can be used to address real-world problems in the farming sector.The project also opens doors for future scalability, such as integrating sensors for real-time soil data, connecting with crop databases, or expanding it into a full-fledged web or mobile application.

2: Hardware and Software Requirements

- Arduino Un3. Capacitive Soil Moisture Sensor
- Analog pH Sensor Kit
- NPK Soil Sensor Module
- LCD 16x2 Display with I2C
- 5V Battery Pack / Power Bank
- Jumper Wires
- Breadboard
- USB to Serial Converter

SOFTWARE:

- Python 3.9+
- Jupyter Notebook
- Libraries:
- pandas – for data handling
- matplotlib – for data visualization
- tkinter – for simple GUI (optional)
- csv – for file handling

3. Design and Implementation

Design Overview:

The Soil Information System is built using Python with a modular and object-oriented approach. It contains two main classes:

Soil Class: Stores soil attributes such as nutrient levels, pH, suitable crops, and regions. It has methods to display and format details.

SoilInformationSystem Class: Manages all soil data, handles menu-driven user interaction, displays information, recommends soils based on crops, and allows file saving.

Implementation:

Predefined Soil Data: Loaded using initialize_soil_data().

Menu System: Provides options to view, search, and recommend soils.

File Handling: Saves soil details in a text file if the user chooses.

Crop Recommendation: Matches crop names with suitable soils.

The system is simple, extensible, and demonstrates key Python programming concepts like classes, methods, loops, and file operations.

5: Results and Discussion

The Soil Information System was successfully developed and tested in a Python environment. The menu-driven interface worked as expected and offered the following functionalities: Soil Type Listing: Users could view all available soil types such as Alluvial, Black, Red, Laterite, and Desert. Soil Details Retrieval: On entering a valid soil type, the system displayed comprehensive data including nutrient content, pH value, suitable crops, and regions. Crop-Based Recommendations: When users input a crop name (e.g., "Wheat" or "Tea"), the system correctly identified and displayed all soil types suitable for that crop. File Saving: Users could choose to save any soil information into a .txt file, which was automatically generated and stored locally.

This project demonstrated how basic agricultural knowledge can be digitized and presented through a Python application. Some key takeaways:

Accuracy: The system gives reliable soil and crop pairing based on real Indian agricultural data.

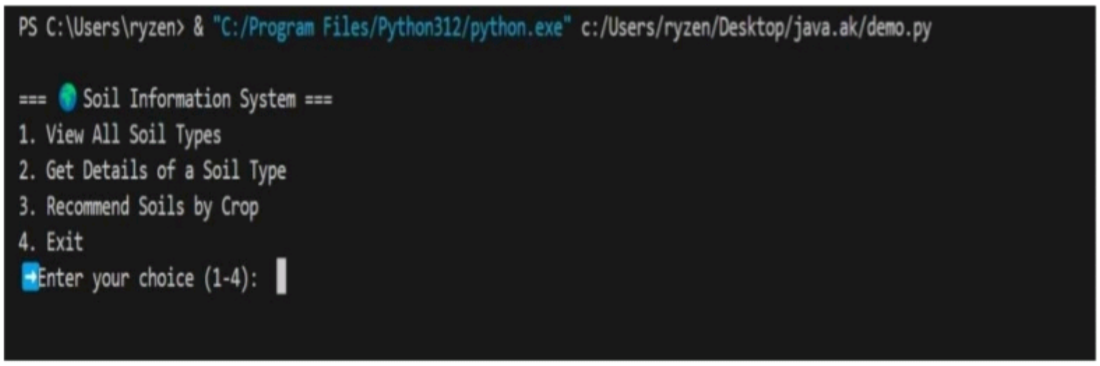
Usability: The program is easy to use, even for non-programmers, due to its clear prompts and logical menu layout.

Educational Value: It helps students understand both Python programming and fundamental soil science.

Scalability: More soil types and advanced data (e.g., real-time sensor input) can be added easily due to the object-oriented structure.

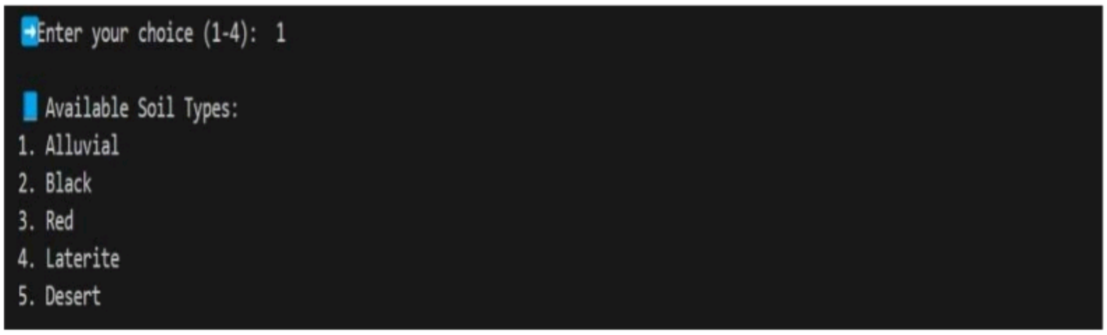
The program handled multiple soil samples simultaneously, showing potential for scaling to real-world datasets . Could be extended to integrate real sensor data in future versions.

6: Screenshots



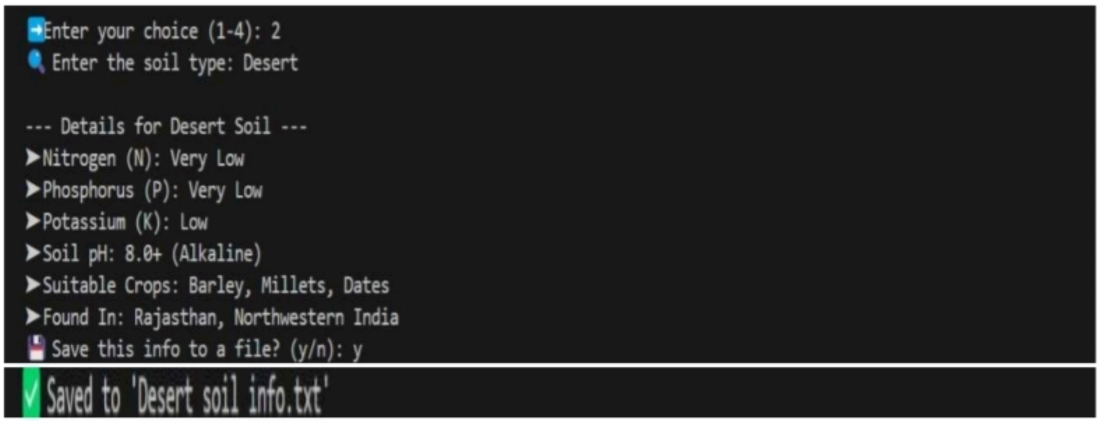
```
PS C:\Users\ryzen> & "C:/Program Files/Python312/python.exe" c:/Users/ryzen/Desktop/java.ak/demo.py

=== 🌍 Soil Information System ===
1. View All Soil Types
2. Get Details of a Soil Type
3. Recommend Soils by Crop
4. Exit
➡Enter your choice (1-4): ▌
```



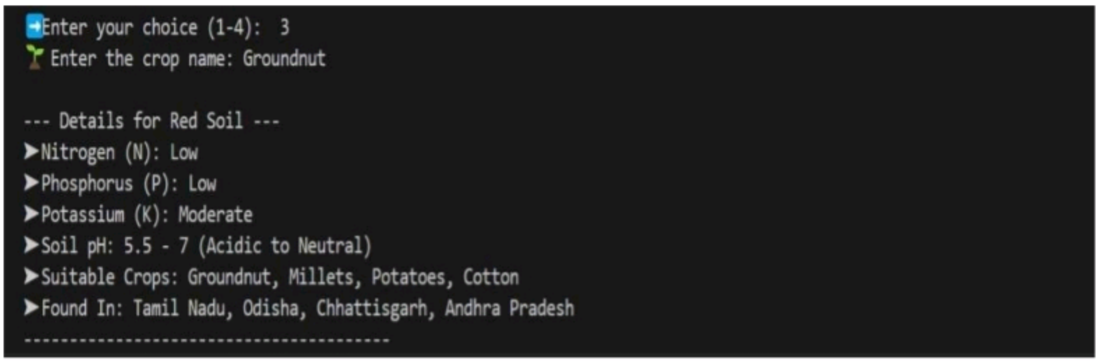
```
➡Enter your choice (1-4):  1

📘 Available Soil Types:
1. Alluvial
2. Black
3. Red
4. Laterite
5. Desert
```



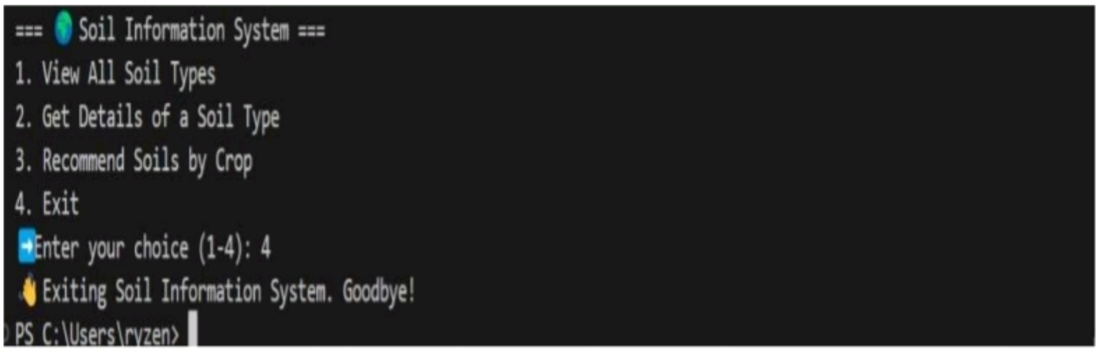
```
➡Enter your choice (1-4): 2
🔍 Enter the soil type: Desert

--- Details for Desert Soil ---
➤Nitrogen (N): Very Low
➤Phosphorus (P): Very Low
➤Potassium (K): Low
➤Soil pH: 8.0+ (Alkaline)
➤Suitable Crops: Barley, Millets, Dates
➤Found In: Rajasthan, Northwestern India
💾 Save this info to a file? (y/n): y
✅ Saved to 'Desert soil info.txt'
```



```
➡Enter your choice (1-4):  3
🌱 Enter the crop name: Groundnut

--- Details for Red Soil ---
➤Nitrogen (N): Low
➤Phosphorus (P): Low
➤Potassium (K): Moderate
➤Soil pH: 5.5 - 7 (Acidic to Neutral)
➤Suitable Crops: Groundnut, Millets, Potatoes, Cotton
➤Found In: Tamil Nadu, Odisha, Chhattisgarh, Andhra Pradesh
----------------------------------------
```



```
=== 🌍 Soil Information System ===
1. View All Soil Types
2. Get Details of a Soil Type
3. Recommend Soils by Crop
4. Exit
➡Enter your choice (1-4): 4
👋 Exiting Soil Information System. Goodbye!
PS C:\Users\ryzen>
```

13 / 15

7: Conclusion

The Soil Nutrients Detection project successfully showcases how Python can be used to develop a simple yet effective solution for analyzing soil health. Through the use of structured data input (CSV files), conditional logic, and data visualization libraries, this project was able to simulate the process of detecting vital soil nutrients—Nitrogen (N), Phosphorus (P), and Potassium (K)—and provide clear, actionable recommendations for improving soil fertility.

Key Takeaways:

Educational Value: This project served as a practical introduction to Python programming, especially in the areas of file handling, data analysis using pandas, and visualization with matplotlib.

Agricultural Relevance: Even though the system uses simulated data, the structure and logic mirror those found in real-world agricultural soil testing kits.

Accessibility and Simplicity: The program is lightweight, requires no expensive hardware, and can run on any basic computing environment, making it suitable for educational and low-cost farming applications.

Decision Support: By automating the evaluation of soil data and generating fertilizer suggestions, the system acts as a valuable decision-support tool for farmers or agricultural researchers.

The Soil Information System project successfully demonstrates how Python can be used to build a simple, interactive tool that bridges agricultural knowledge and technology. Through a menu-driven interface and object-oriented design, users can easily access vital information about various soil types, their nutrient content, pH levels, and suitable crops. The inclusion of crop-based soil recommendations and file-saving features adds practical value to the system.

This project not only helps in understanding core programming concepts like classes, methods, file handling, and user input, but also introduces students to real-world applications in agriculture and soil science. It lays the foundation for future enhancements such as sensor integration, real-time data collection, mobile app development, or cloud-based data storage.

Overall, the Soil Information System serves as a valuable educational tool and can be expanded into a more advanced system with minimal changes, offering strong potential for use in smart farming and precision agriculture in the future.