KING'S OWN INSTITUTE*

Success in Higher Education

INDIVIDUAL ASSIGNMENT COVERSHEET

Family Name: Ahnaf Given Name: Shakil

Student Number: 20031759 Lecturer's Name: Rabia Bashir mam

Subject Name: Programming Fundamentals (ICT 701)

Assignment Title: Smart Agriculture Monitoring System (SAMS)

Declaration

(This declaration must be completed by the student or the assignment will not be marked.)

I certify the following:

- I have read and understood the *Student Academic Misconduct Policy*
- This assignment is my own work based on my personal study and or research.
- I have acknowledged all material and sources used in the preparation of this assignment including any material generated in the course of my employment.
- The assignment has not previously been submitted for assessment.
- I have not copied in part or in whole or otherwise plagiarised the work of other students.
- I have read and I understand the criteria used for assessment.
- The assignment is within the word and page limits specified in the unit outline.
- The use of any material in this assignment does not infringe the intellectual property / copyright of a third party.
- I understand that this assignment may undergo electronic detection for plagiarism, and an anonymous copy of the assignment may be retained on the database and used to make comparisons with other assignments in future.
- By completing this coversheet in full and submitting this assignment electronically, I am bound by the conditions of the KOI's *Student Academic Misconduct Policy* and the declaration on this coversheet.

Ahnaf Shakil

Signature

30 / 09/ 2025

Date

Assignment Receipt

Family Name: Ahnaf Given Name:(s) Shakil

Student Number: 20031759 Lecturer's Name: Rabia Bashir mam

Subject Name: Programming Fundamentals (ICT 701)

Assignment Title: Smart Agriculture Monitoring System (SAMS)

Ahnaf Shakil

Signature

30 / 09/ 2025

Date

This report provides a comprehensive analysis of the Smart Agriculture Monitoring System (SAMS), evaluating its development through two distinct architectural lenses: a Graphical User Interface (GUI) and a Command-Line Interface (CLI). The system is designed to bridge the gap between traditional farming and modern data-driven management.

1. Executive Summary

The Smart Agriculture Monitoring System (SAMS) is a software solution developed to streamline agricultural operations. It manages four critical pillars: human resources (farmers and technicians), operational efficiency (task scheduling), environmental intelligence (field observations), and financial sustainability (transaction tracking). By implementing this system in both GUI and text-based formats, the project demonstrates how software can adapt to different user environments from high end office workstations to low resource remote terminals.

2. System Architecture and Data Modeling

At the core of SAMS is an Object Oriented Programming (OOP) structure. The system is built using Python, leveraging its ability to handle complex data structures with ease.

2.1 Core Entities

The system is modeled around four primary classes:

Farmer Class: Stores the unique Identity (ID), name, and farming type (organic, livestock, crop). It also acts as a container for "Observations," allowing for a longitudinal record of land health.

Technician Class: Manages the technical workforce, recording their specialization (irrigation expert, soil chemist).

Task Class: A dynamic class that tracks agricultural activities, linking descriptions and dates to specific technicians, with a state-machine logic for status (Pending vs. Completed).

Transaction Class: Ensures financial transparency by logging payments and revenue associated with specific farmers.

2.2 Data Persistence

A critical requirement was the ability to retain data after the program closes. The system utilizes Pickle Serialization.

Mechanism: The SAMS object, which contains all lists of farmers, tasks, and financials, is converted into a byte stream and saved to sams_data.pkl.

Efficiency: This ensures that upon restarting either the GUI or the text based version, the user's previous work is immediately available.

3. The Graphical User Interface (GUI) Analysis

The GUI version was developed using the Tkinter library, focused on user experience (UX) and visual hierarchy.

3.1 Visual Design and Branding

The interface utilizes a professional aesthetic, incorporating a background image located at F:\Assignment 04 (Python)\background.jpg. The use of the PIL (Pillow) library allows for high quality image scaling, ensuring the "Smart Agriculture" theme is consistently presented.

3.2 User Interaction Flow

The GUI simplifies complex navigation through a "Windowed" approach. Each management sector opens in a Toplevel popup window. This prevents the user from being overwhelmed by too many input fields on a single screen. Key features include:

Event Handling: Hover effects on buttons (changing colors from light blue to dark blue) provide tactile feedback.

Input Validation: The GUI uses messagebox alerts to warn users about duplicate IDs or empty fields, preventing data corruption at the entry point.

4. The Text-Based (CLI) Interface Analysis

The text based version was engineered for speed, stability, and compatibility. It operates within a standard terminal using a loop driven menu system.

4.1 Logic and Robustness

The CLI version prioritizes robustness. One of the primary challenges in text interfaces is "user straying" where a user types an invalid character.

Input Sanitization: The system uses conditional loops. If a user enters "9" in a 6-option menu, the system triggers a warning: [!] INVALID CHOICE.

Navigation: Each sub-menu includes a "Back to Main Menu" option, ensuring a non-linear navigation path that mimics the flexibility of a GUI.

4.2 Information Density

Unlike the GUI, which hides details behind buttons, the CLI version focuses on scannability. When viewing records, the system prints formatted strings that display every attribute of the object (ID, Name, Specialization, etc.) in a single line, making it easier for power users to audit large amounts of data quickly.

5. Financial Management and Economic Tracking

The financial module of the SAMS software serves as the economic backbone of the system, designed to ensure fiscal transparency and accountability between the management and the farmers. Within the text based interface, this section operates through a structured sub-menu that allows for the recording of unique transaction IDs, the associated farmer's name, and the specific monetary amount involved. To maintain data integrity, the system implements a strict numeric validation check, if a user enters non numeric characters for an amount, the system provides an immediate text based warning and prompts for a re entry rather than allowing the application to crash.

Furthermore, the module is programmed to aggregate all recorded payments to provide an instantaneous "Total Revenue" calculation, which is essential for assessing the overall health of the agricultural enterprise. Every transaction is cross referenced against the existing database to prevent duplicate entries, ensuring that each financial record is distinct and auditable. This transition from simple data storage to an active financial monitoring tool allows administrators to track income trends and verify payments without requiring external accounting software.

6. Critical Implementation Features

Two specific requirements were prioritized in both versions: Duplicate ID Prevention and Full Detail Display.

6.1 Duplicate ID Prevention

In any database style system, the Unique Identifier (ID) is the primary key. If two farmers share the same ID, data retrieval becomes ambiguous.

Logic: Before a new object is appended to the system list, a "list comprehension" or "any()" check is performed.

Code Implementation: if any(f.id == new_id for f in system.farmers):

Result: This logic maintains the integrity of the data, ensuring that "Farmer 101" always refers to a single, specific entity.

6.2 Full Attribute Transparency

Earlier iterations of the software only showed IDs and Names. The final versions were updated to ensure that "Specialization" (for Technicians) and "Type" (for Farmers) are always visible during the "View" command. This is vital for decision-making; for instance, a manager needs to see the technician's specialization to assign the correct task.

7. Future Enhancements

While the current SAMS provides a solid foundation, several features could be added to move toward a "Version 2.0":

Search and Filter: Adding a search bar in the GUI or a search command in the CLI to filter farmers by type or technicians by specialization.

CSV Export: Allowing users to export the financial transactions to a spreadsheet format (CSV or Excel) for accounting purposes.

Real time Sensors: Integrating the "Observations" module with actual IoT sensors to automatically log soil moisture and temperature.

8. Conclusion

The development of the Smart Agriculture Monitoring System illustrates the versatility of Python in solving real world problems. Whether through the visually engaging Tkinter GUI or the streamlined, error resistant Text Based CLI, the system successfully manages agricultural data with high integrity. By enforcing unique IDs and ensuring data persistence via Pickle, SAMS provides a reliable tool for modernizing farm management. The dual mode development ensures that the system is ready for any deployment scenario, balancing user friendliness with technical efficiency.