# STUDENT ATTENDENCE MANAGEMENT SYSYTEM

## Submitted By:

Name: Rohan Jaria
Registration No.: 25BCE10437
Course:CSE1021
College: VIT Bhopal University

# ***Title of the Project***

"Student Attendance Management System using Python and Tkinter"

# Introduction

Attendance recording is an important part of classroom management. Traditional methods such as manual registers are time-consuming, prone to errors, and difficult to maintain. Digital attendance systems provide an efficient alternative for recording and managing attendance.

This project focuses on creating a simple yet functional Student Attendance Management System using Python's Tkinter library. It allows users to enter a student's name, roll number, and attendance status. The application automatically records the date and time of marking attendance. The attendance can be viewed, modified, and deleted directly through the GUI.

# Problem Statement

**Manual attendance systems require paperwork, increase the chances of errors, and are not effective for long-term use. Teachers need a system that:**

01. Reduces manual errors

02. Stores records systematically

03. Saves time

04. Provides a clear interface for daily attendance management

**Thus, a digital solution is required for efficient attendance monitoring.**

# Objectives

- To develop an easy-to-use GUI-based attendance system.
- To record student attendance along with date and time.
- To allow insertion, deletion, and viewing of attendance data.
- To ensure error-free and fast attendance marking.

# Functional Requirements

1. **User Authentication (Optional) –** Basic user login before accessing the system.
2. **Add Student Attendance –** Allow user to enter name and roll number.
3. **Mark Present –** Update student status as "Present" with timestamp.
4. **Mark Absent –** Update student status as "Absent" with timestamp.
5. **View Attendance Records –** Display all attendance information in a table.
6. **Select Student Record –** Automatically load selected row details into input fields.
7. **Delete Student Attendance Record –** Remove a specific student's attendance.
8. **Auto Timestamping –** System must automatically add date & time.

# Non-Functional Requirements

1. **Performance –** The GUI must load quickly and respond instantly to button clicks.
2. **Usability –** Interface must be simple, clear, and easy to operate.
3. **Reliability –** Attendance records should update correctly without crashes.
4. **Portability –** Application should run on any OS with Python installed.
5. **Maintainability –** Code should be readable and easy to update.
6. **Scalability (Limited) –** System can handle small classroom datasets efficiently.
7. **Security (Basic) –** No unauthorized user should modify records; optional login can be added.

# CRUD Operations Slide

**C – Create:** Add Student Attendance

**R – Read:** View Attendance Table

**U – Update:** Mark Present / Mark Absent

**D – Delete:** Remove Attendance Record

# Technologies Used

## 01

### Hardware Requirements

- Minimum 4GB RAM
- Any Windows/Linux/MacOS machine

## 02

### Software Requirements

- Python 3.x
- Tkinter (Python's built-in GUI library)
- ttk module for treeview table display
- datetime module for timestamping records

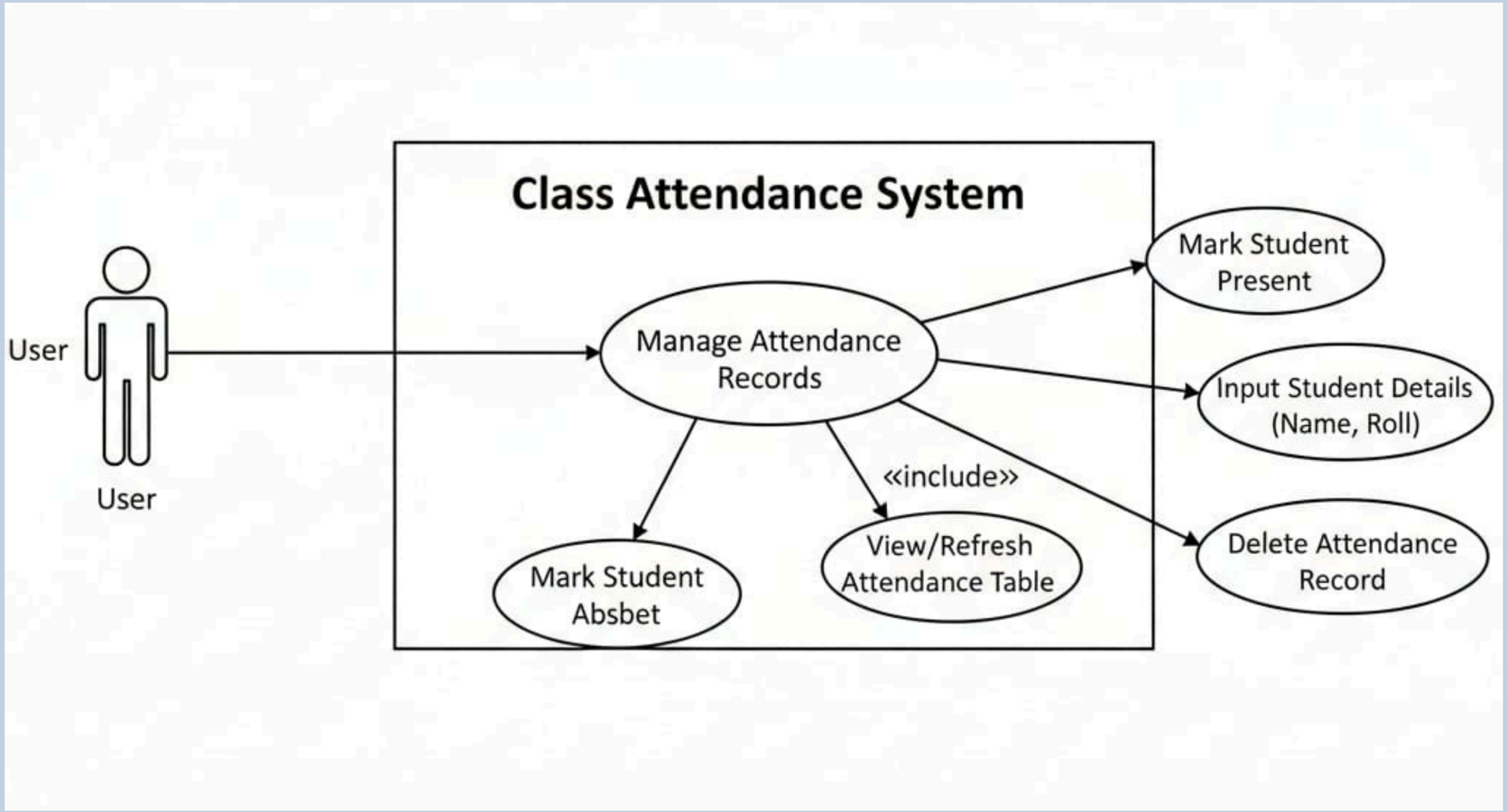
# System Architecture

***The system follows a simple architecture:***

- **User Input Layer –** GUI-based input for name, roll number, and attendance action.
- **Processing Layer –** Handles operations (Add, Mark Present, Mark Absent, Delete).
- **Storage Layer –** Stores attendance in an in-memory dictionary.
- **Output Layer –** Displays results in the GUI table.

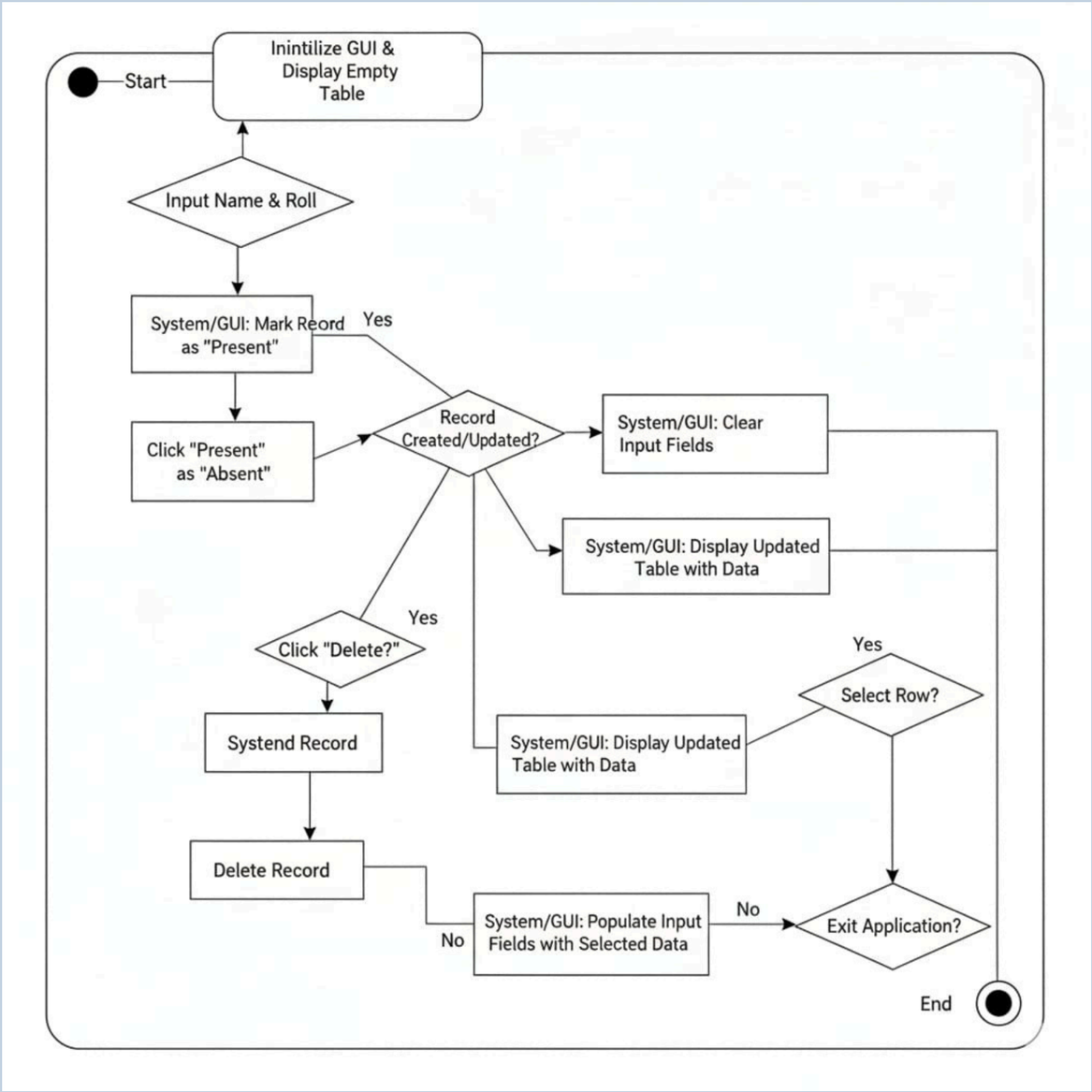
# Design Diagram

## Use case diagram :



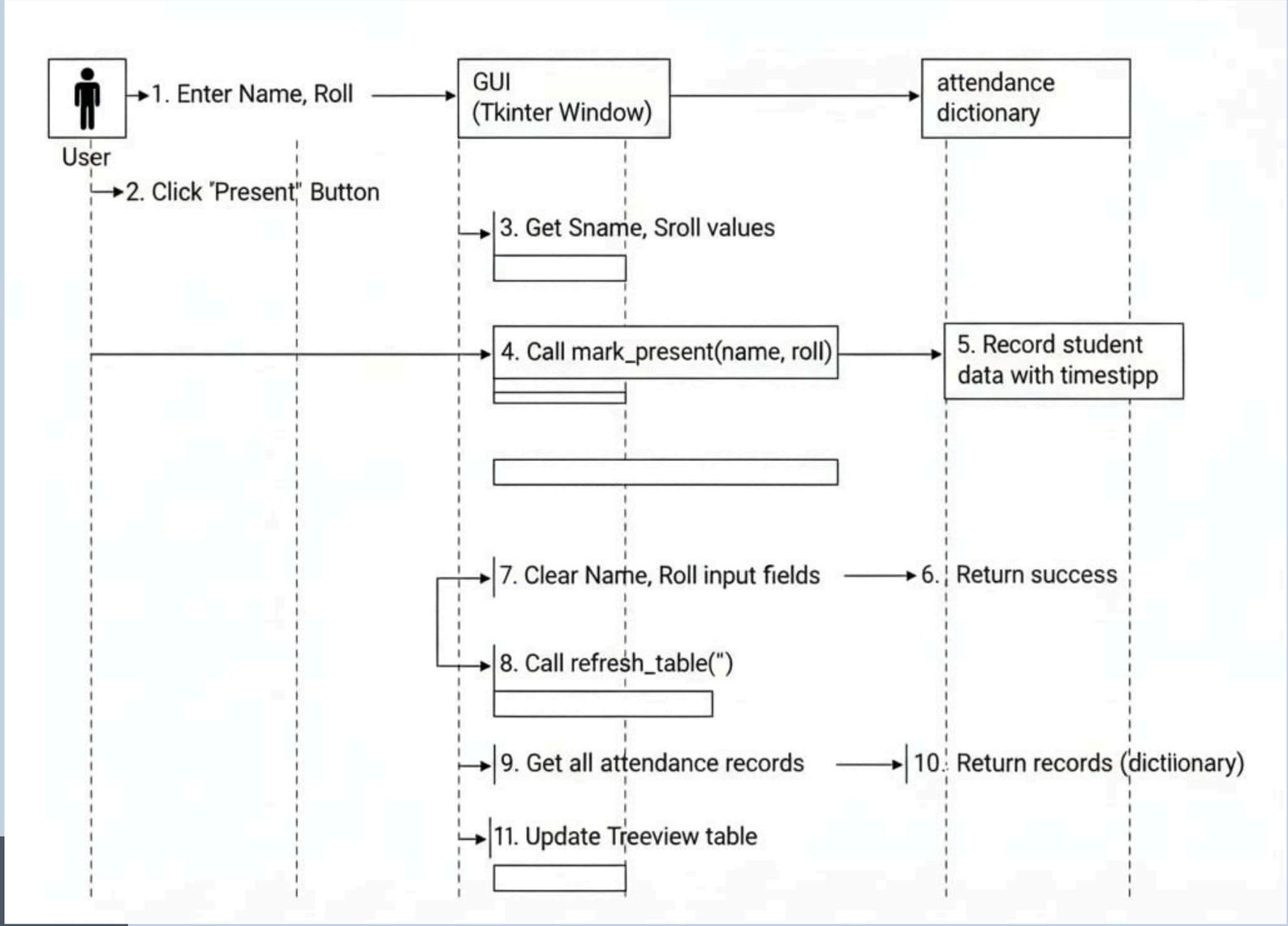
# Design Diagram

## Workflow diagram:



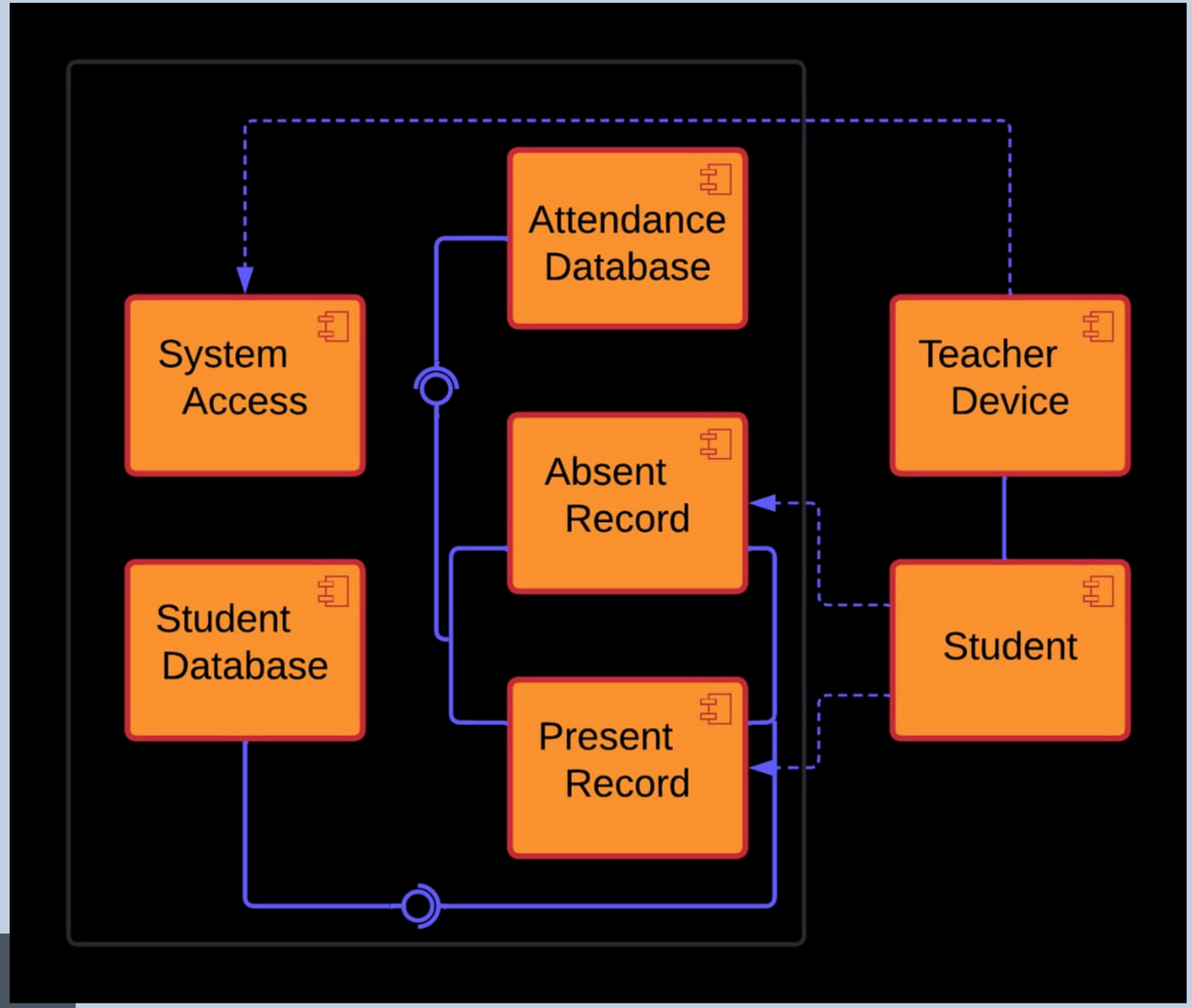
# Design Diagram

## Sequence diagram :



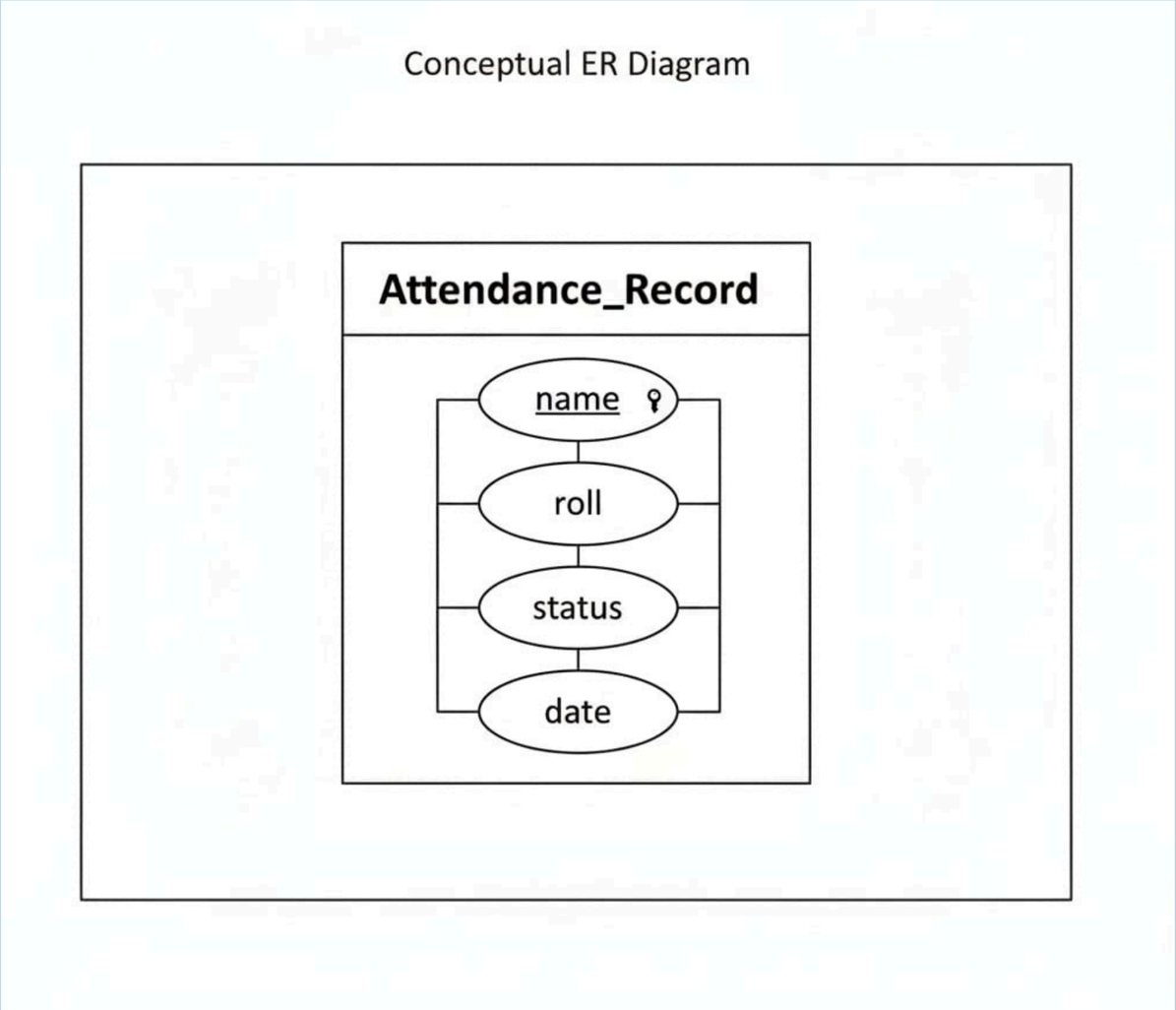
# Design Diagram

## Class component diagram:



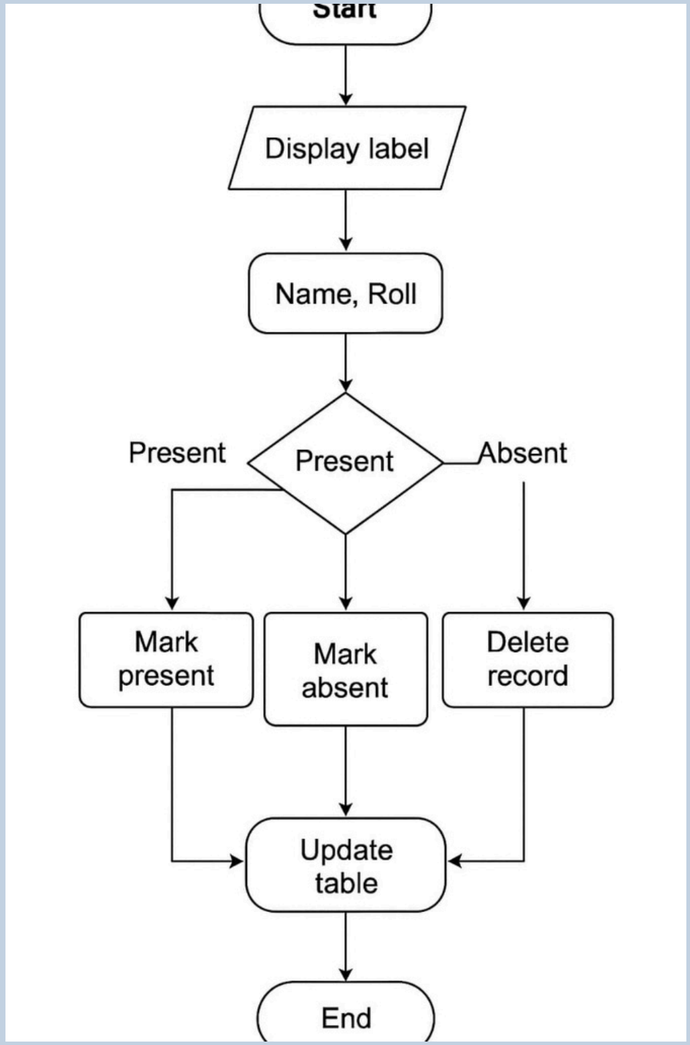
# Design Diagram

## ER Diagram:



# Design Diagram

## Flowchart:



# Design Decisions & Rationale:

- Tkinter was chosen because it is lightweight, built-in, and easy for rapid GUI development.
- Dictionary-based storage was used for simple and fast CRUD operations.
- Treeview widget was implemented to display attendance in a structured table format.
- StringVar() and IntVar() were used for dynamic linking between GUI fields and variables.
- Modular function design (mark_present, mark_absent, delete, refresh_table) improves readability and maintenance.
- Automatic timestamping ensures each attendance entry is recorded with real-time date and time.
- Row selection binding allows records to be auto-filled into input fields for review or deletion.

# Implementation Details:

- **Programming Language:** Python
- **GUI Framework:** Tkinter
- **Widgets Used:** Labels, Entry, Buttons, Frames, Treeview, Scrollbar
- **Data Storage:** In-memory dictionary

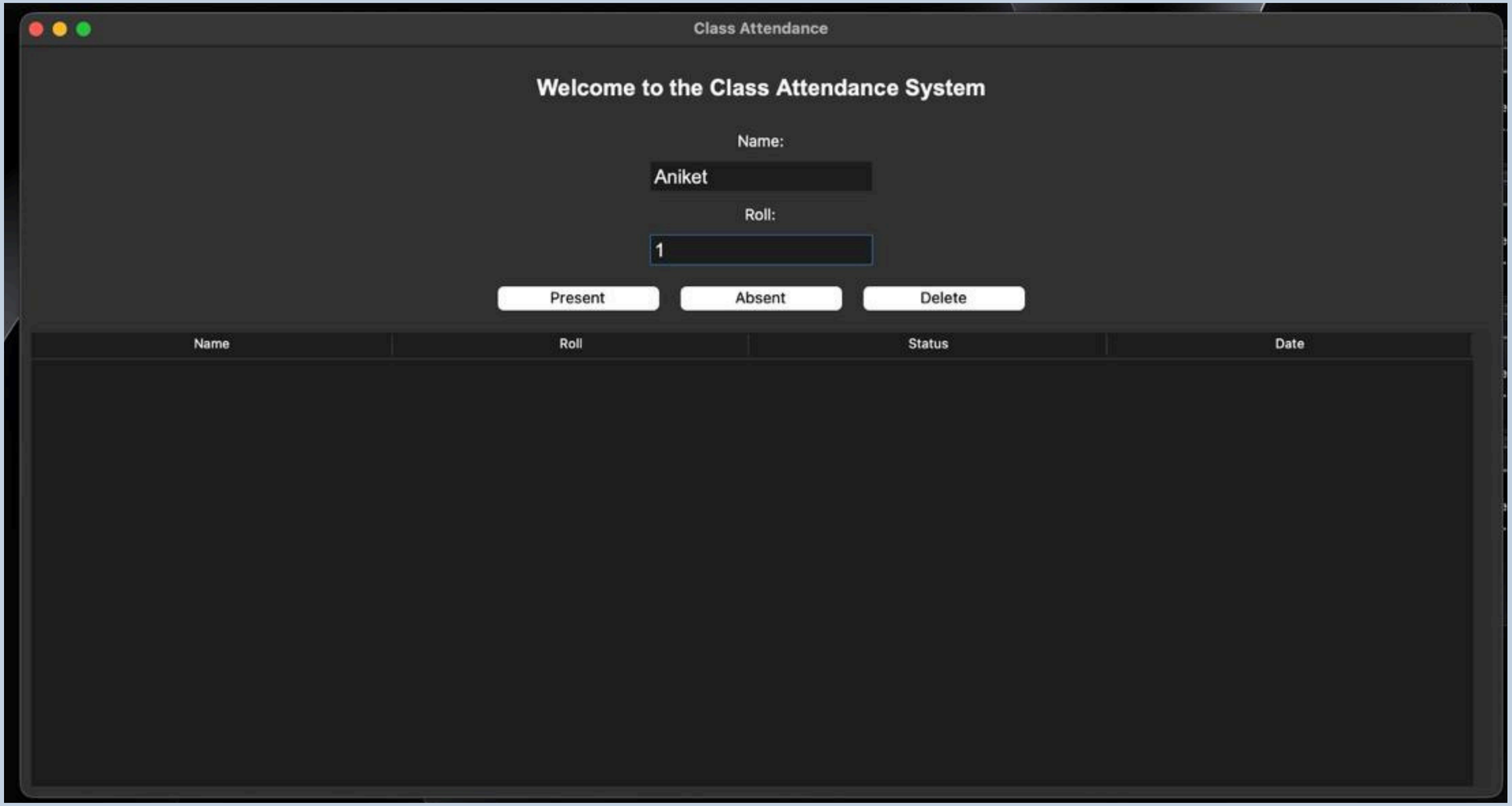
## Core Features:

- Name and Roll number input
- Mark Present/Absent
- Delete record
- Auto-refreshing attendance table
- Real-time timestamps
- Row selection for easy editing

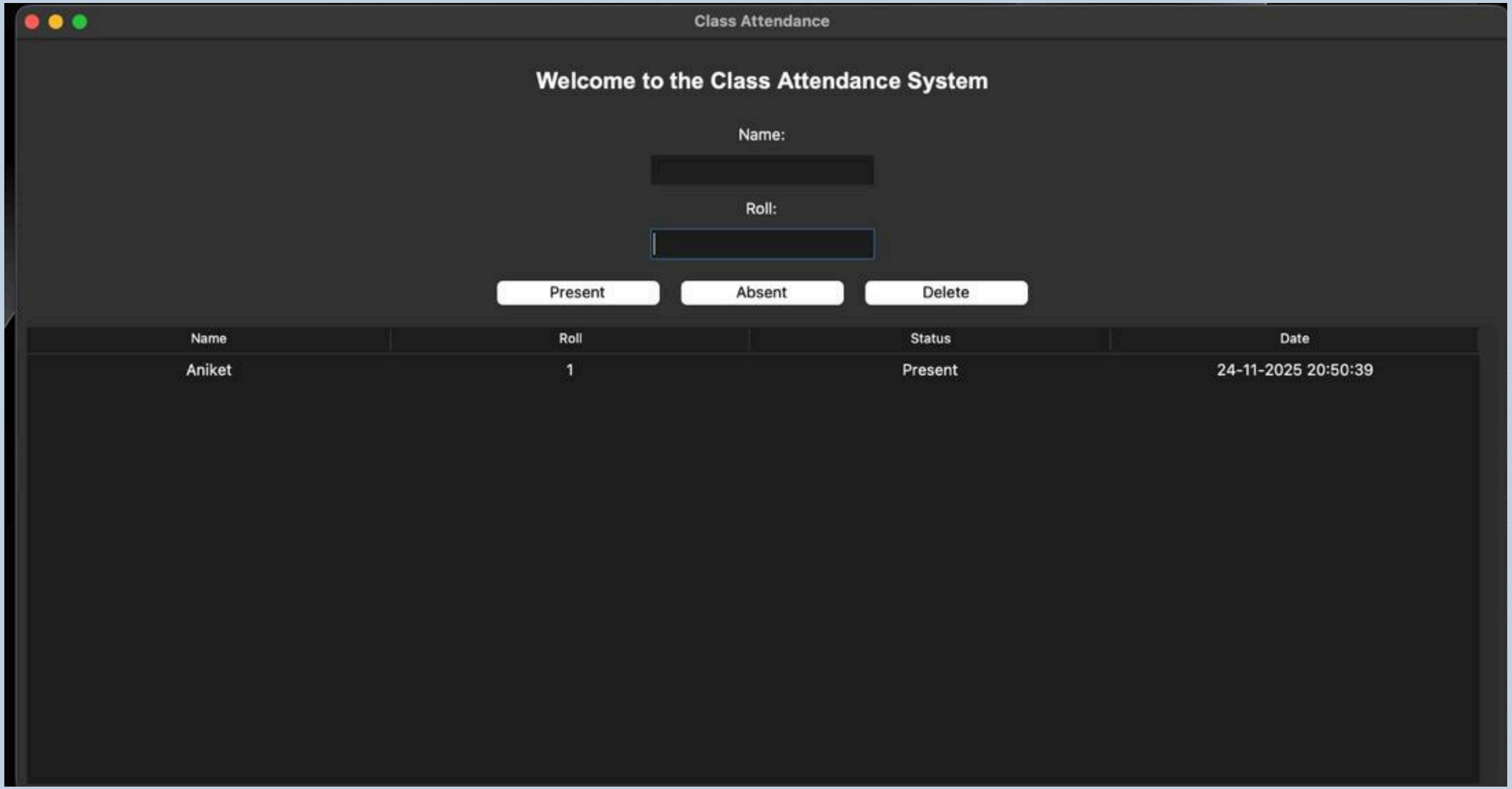
## Workflow Steps:

1. User enters student name & roll number
2. Clicks Present/Absent → record added
3. Table refreshes and displays updated data
4. Selecting a row auto-fills input fields
5. Delete button removes the selected record

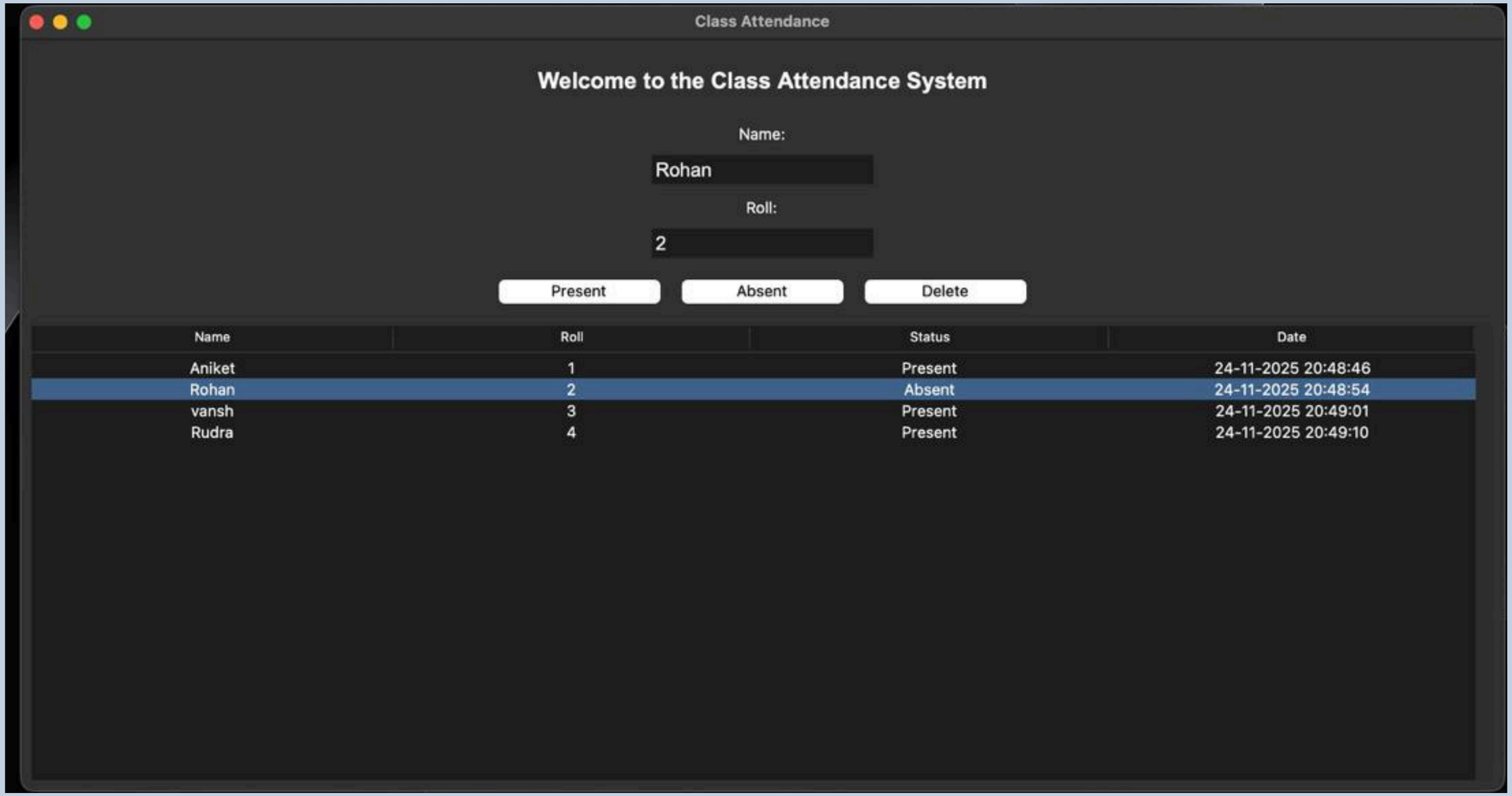
# Screenshots/Results:



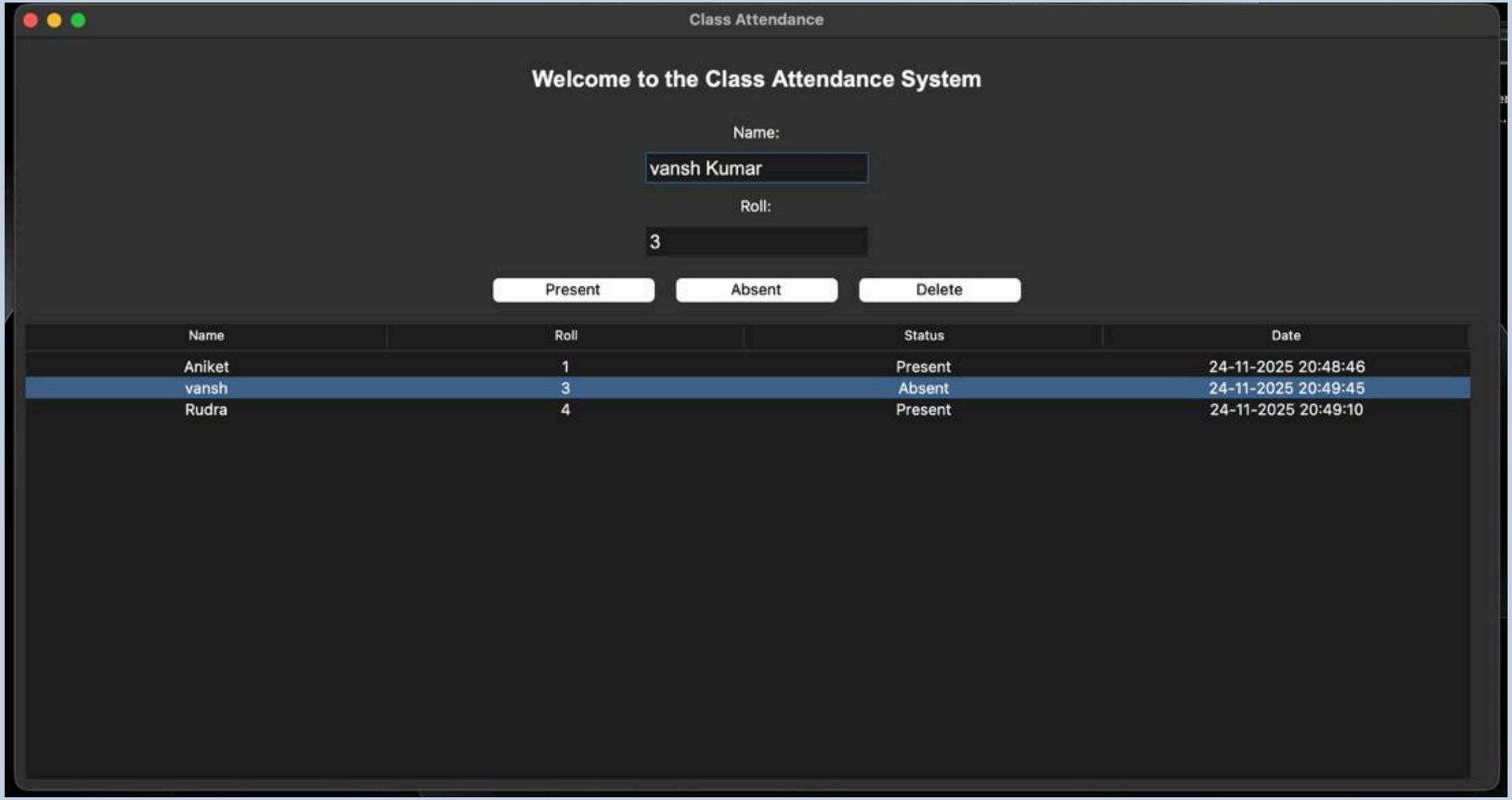
# Screenshots/Results:



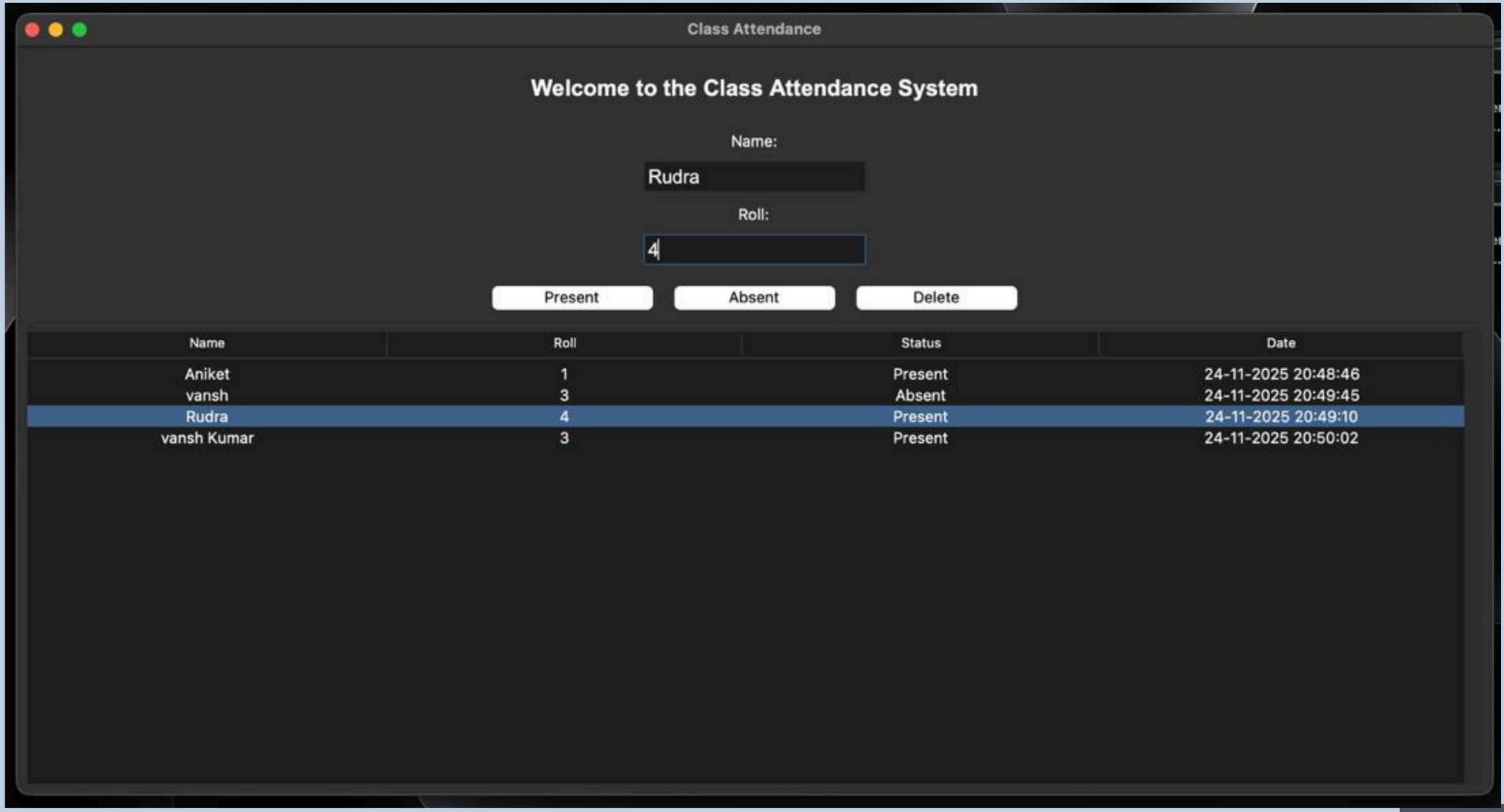
# Screenshots/Results:



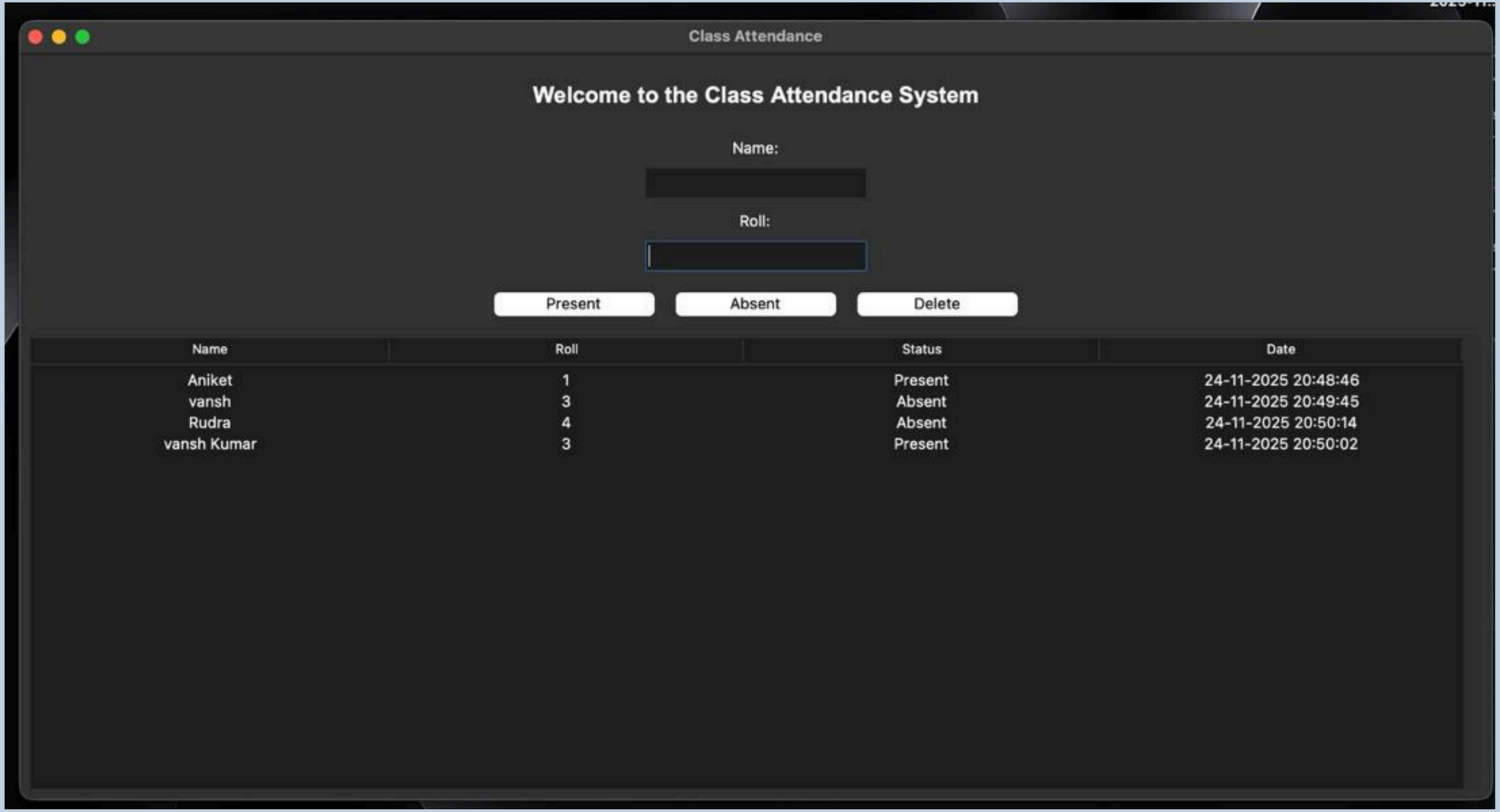
# Screenshots/Results:



# Screenshots/Results:



# Screenshots/Results:



# Testing Approach:

- **Unit Testing:**
  - Tested functions like marking present, marking absent, deletion, and table refresh.
- **GUI Interaction Testing:**
  - Validation of input fields
  - Button responsiveness
  - Accurate table refresh
  - Row selection update
- **Boundary Testing:**
  - Empty fields
  - Invalid roll number
  - Duplicate record entries
- **Usability Testing:**
  - Layout alignment
  - User navigation and clarity

# Challenges Faced:

- Keeping the table (Treeview) synced with the dictionary data
- Implementing smooth row selection and auto-fill
- Handling invalid or incomplete input
- Integrating scrollbars properly with Treeview
- Managing GUI refresh efficiently

# Learnings & Key Takeaways

- Understanding of Tkinter basics and widget usage
- Practical implementation of CRUD operations in GUI
- Experience creating table-based UI using Treeview
- Improved knowledge of event binding and callbacks
- Importance of writing modular and clean code
- Handling timestamps using datetime

# Future Enhancements:

- Add database support (SQLite/MySQL) for permanent storage
- Add CSV/Excel export feature
- Introduce Update/Edit button
- Add login system (Teacher/Admin)
- Support multiple classes/sections
- Add attendance percentage calculation
- Improve UI using themes (e.g., ttkbootstrap)

# References


- Python Official Documentation
- Tkinter Docs
- Treeview Widget Documentation
- StackOverflow (for debugging references)
- GeeksForGeeks & W3Schools (GUI examples)

# CONCLUSION

The Student Attendance Management System successfully provides a simple and user-friendly interface for marking and maintaining attendance records. It reduces human errors, saves time, and helps in digitalizing classroom attendance systems. The project demonstrates Python GUI development skills using Tkinter and highlights basic data management concepts.

# THANK YOU