PROJECT REPORT

TITLE :

TIME TABLE CRUD MANAGER

COURSE	Introduction to Problem-solving and Programming -CSE1021
NAME	AKSHARA SHARMA
REGISTRATION NO.	25BCE10666
FACULTY	DR.MANI MARAN
INSTITUTION	VIT BHOPAL
DATE OF SUBMISSION	25-11-2025

TABLE OF CONTENT

1. Introduction

The **Timetable CRUD Manager** is a simple and easy-to-use program designed to help people organize their schedules without relying on complicated software or handwritten notes. Many people struggle to keep track of their daily activities, appointments, classes, or tasks, especially when their schedules change frequently. This tool solves that problem by giving you a clear and flexible way to manage your timetable using only simple text commands.

At its core, the program allows you to **Create**, **Read**, **Update**, and **Delete** timetable entries — these actions are commonly known as CRUD operations. In everyday terms, this means you can add new schedule items whenever something comes up, look at the list of everything you have planned, change an existing entry if the time or details shift, and remove an entry when you no longer need it. All of these actions are carried out through a straightforward command-line interface, which makes the tool lightweight, fast, and accessible even for beginners.

The Timetable CRUD Manager is useful for students who want to organize their classes, workers who need to track deadlines or shifts, or anyone who simply wants a tidy and dependable way to manage their time. Because it stores information in a structured and organized manner, it becomes much easier to review and adjust your schedule whenever necessary. Overall, this program provides a practical and user-friendly solution for keeping your timetable accurate, up to date, and easy to manage.

2.Problem Statement

Managing a personal or academic timetable can be challenging, especially when schedules change often or when multiple activities need to be tracked throughout the day. Many people rely on handwritten notes, scattered digital files, or memory to manage their daily tasks, classes, or appointments. These methods can easily lead to confusion, forgotten events, or overlapping commitments. Simple mistakes—like losing a note or forgetting to update a changed time—can cause unnecessary stress and disorganization.

There is a need for a simple, accessible, and reliable way to store, view, edit, and remove timetable entries without requiring advanced software or technical expertise.

Most scheduling tools are either too complex, too heavy, or require a graphical interface, which may not always be available.

The **Timetable CRUD Manager** aims to solve this problem by providing a lightweight, command-line-based system that allows users to efficiently manage timetable entries. By offering basic yet essential operations—creating, reading, updating, and deleting—the tool ensures that users can keep their schedules organized, up-to-date, and easy to review. This helps reduce confusion, improves time management, and provides a straightforward solution for anyone who needs a simple way to handle schedule-related tasks.

3.Functional Requirements

1. Add New Timetable Entry

The system must allow the user to create a new timetable entry.
The user should be able to enter details such as the event name, date, time, and any additional notes.
The entry must be saved and stored for future access.

2. View All Timetable Entries

The system must display a list of all saved timetable entries.
Each entry should show key information, making it easy for the user to see their entire schedule at a glance.

3. View Details of a Specific Entry

The system must allow the user to select a specific timetable entry and view detailed information about it.
This includes showing the full description, date, time, and any notes.

4. Update an Existing Timetable Entry

The system must allow the user to edit or modify an existing entry.
Users should be able to change any detail of an entry, such as time, description, or date.
The updates must be saved correctly.

5. Delete a Timetable Entry

The system must allow the user to remove an entry from the timetable.
The system should confirm the deletion to avoid accidental removal.

4.Non-Functional Requirements

1. Usability

The system should be easy to use, even for people who are not familiar with technical tools.
The menu options and instructions should be clear, simple, and easy to understand.

2. Performance

The system should respond quickly to user commands.
Adding, viewing, updating, or deleting entries should take only a fraction of a second.

3. Reliability

The system should consistently store and retrieve timetable entries without losing data.
It should perform correctly every time a user interacts with it.
No unexpected crashes or errors should occur during normal use.

4. Data Integrity

All data must remain accurate and unchanged unless the user updates or deletes it.
The system should prevent invalid or partial data from being saved.

5. Portability

The system should be able to run on different devices and operating systems that support the required programming language (example: Windows, macOS, Linux).
It should not depend on hardware-specific features.

5.System Architecture

The **Timetable CRUD Manager** follows a simple and lightweight architecture designed to make timetable management easy and reliable. The system is usually divided into three main layers: the **User Interface**, the **Application Logic**, and the **Data Storage Layer**.

1. User Interface (CLI Layer)

This is the part the user interacts with. It displays menus, asks for input, and shows messages or results.

Responsibilities:

- Show options to the user
- Collect user input

2. Application Logic (Manager Layer)

This is the “brain” of the system. It receives the user's choices from the interface and decides what to do. It processes the input, checks if the information is valid, and performs the requested operation.

Responsibilities:

- Handle CRUD operations
- Validate user input
- Communicate with the database or storage layer

3. Data Storage Layer (Database/File System)

This layer stores all timetable entries permanently. Depending on your implementation, this may be:

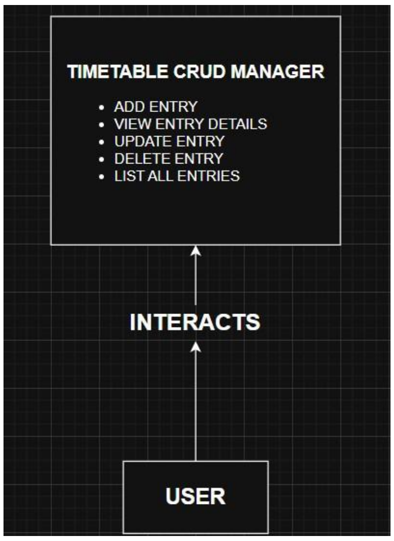
- A text file
- A JSON file
- A local database
- A simple in-memory list (temporary)

Responsibilities:

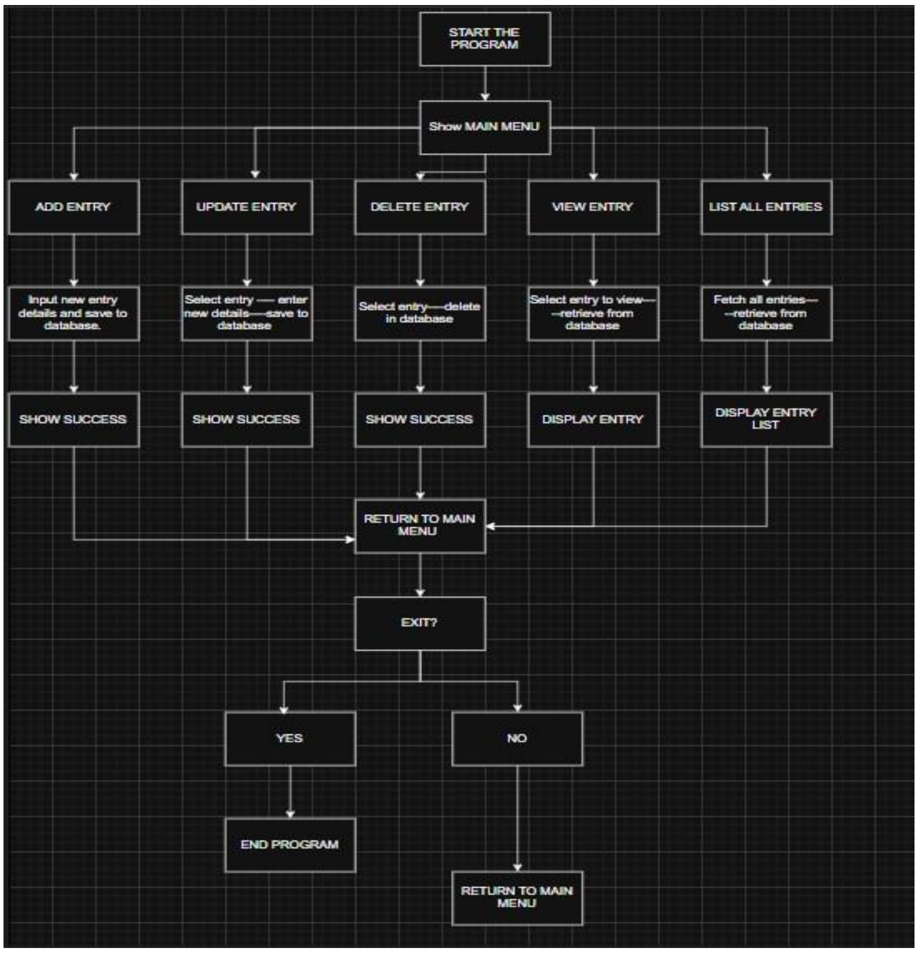
- Save new entries
- Update existing ones
- Delete entries

6.Design Diagrams

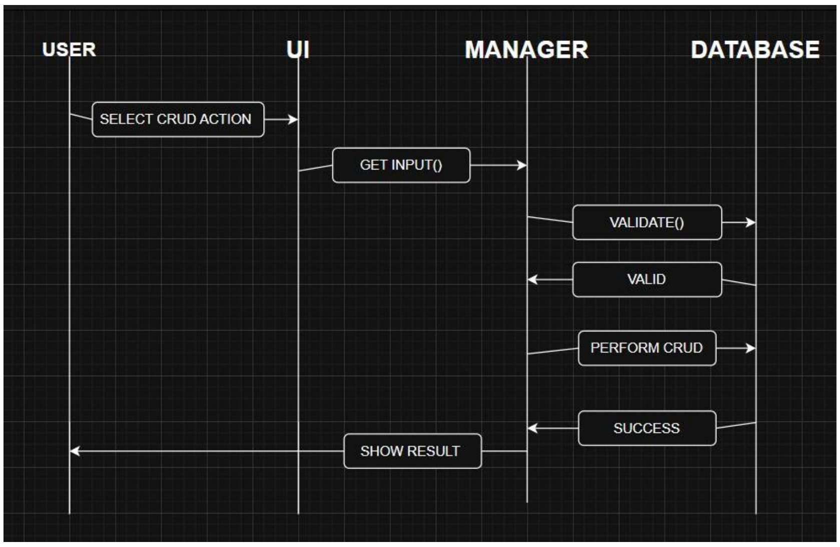
1.Use-case diagram



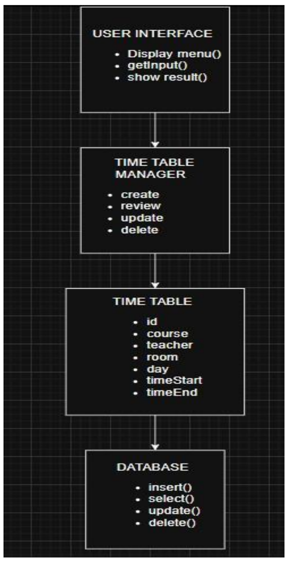
2.Workflow diagram



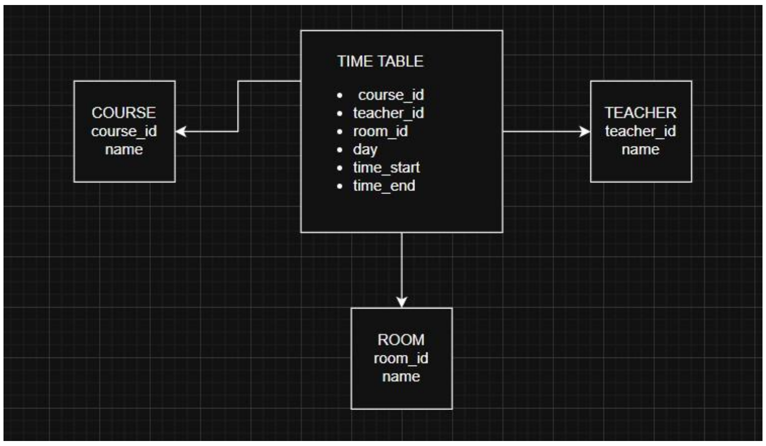
3. Sequence Diagram



4.Component Diagram



5.ER Diagram



7.Design Decisions & Rationale

1. Use of a Command-Line Interface (CLI)

The system uses a text-based command-line interface. A CLI is simple, lightweight, and does not require complex graphical elements. It works on almost any computer without additional setup and is easy for beginners to use. It also keeps the system fast, focusing only on functionality rather than visuals.

2. CRUD-Based Functionality

The system is built around Create, Read, Update, and Delete operations. These are the essential operations needed to manage any kind of record or schedule. Keeping the system focused on CRUD ensures simplicity while covering everything a user needs to maintain a timetable effectively.

3. Layered Architecture (UI, Logic, Storage)

The system is divided into three layers: interface, manager logic, and data storage. This separation makes the system easier to maintain, understand, and expand in the future. Each layer has a clear job—one handles user interaction, one processes tasks, and one stores data. This avoids confusion and keeps the code organized.

4. File-Based or Simple Storage

A simple storage method (like a text file or JSON file) is used instead of a full database.It keeps the project lightweight and easy for beginners to run without extra setup. A file-based approach is enough for managing small to medium sets of timetable entries and does not require installing additional software.

5. Text-Based Menu Navigation

Users interact through a numbered or text-based menu. Menus reduce confusion and make the system more user-friendly. Users don’t need to remember commands—they simply choose an option. This approach is intuitive, especially for users who are new to CLI tools.

6. Validation of User Input

The system checks for incorrect or missing input. This prevents errors and ensures data stays clean. For example, empty fields or invalid formats can cause confusion later. Input validation creates a smoother and safer user experience.

7. Focus on Simplicity Over Complexity

Only essential features are included; no unnecessary complexity. The goal is to build a practical, easy-to-use timetable manager—not a bloated, overly complicated system. Focusing on essential features makes the tool more reliable and approachable.

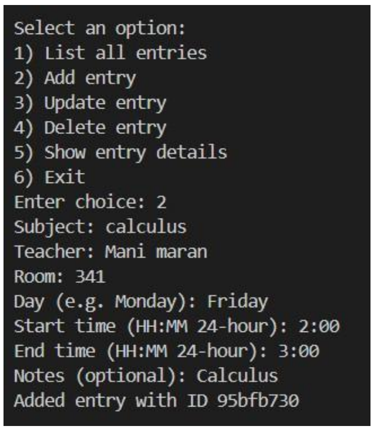
8.Implementation Details

The **Timetable CRUD Manager** is implemented as a command-line application using a structured, layered approach. The program begins by displaying a menu that lets users choose actions such as adding, viewing, updating, or deleting timetable entries. Each option triggers a specific function that handles the corresponding operation.

All core logic—such as validating input, managing entries, and coordinating between components—is handled in the manager module. The system stores timetable data in a simple file-based format (such as a text or JSON file), making it easy to save and retrieve information without requiring a full database. Each entry is stored with key details like title, date, time, and description.

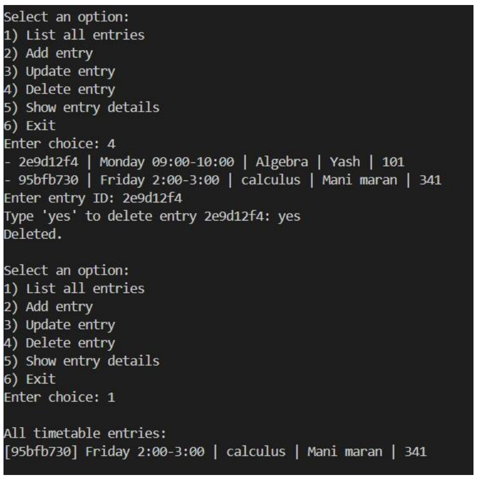
The program uses loops to keep the menu active until the user chooses to exit, ensuring continuous interaction. Error handling is included to manage invalid inputs and prevent crashes. Overall, the implementation focuses on being lightweight, easy to run on any system, and simple for users to understand.

ADD ENTRY



```
Select an option:
1) List all entries
2) Add entry
3) Update entry
4) Delete entry
5) Show entry details
6) Exit
Enter choice: 2
Subject: calculus
Teacher: Mani maran
Room: 341
Day (e.g. Monday): Friday
Start time (HH:MM 24-hour): 2:00
End time (HH:MM 24-hour): 3:00
Notes (optional): Calculus
Added entry with ID 95bfb730
```

DELETE ENTRY

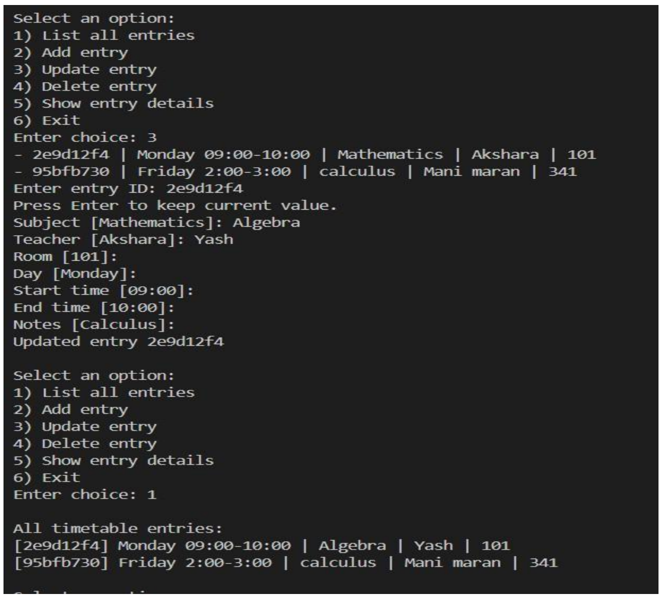


```
Select an option:
1) List all entries
2) Add entry
3) Update entry
4) Delete entry
5) Show entry details
6) Exit
Enter choice: 4
- 2e9d12f4 | Monday 09:00-10:00 | Algebra | Yash | 101
- 95bfb730 | Friday 2:00-3:00 | calculus | Mani maran | 341
Enter entry ID: 2e9d12f4
Type 'yes' to delete entry 2e9d12f4: yes
Deleted.

Select an option:
1) List all entries
2) Add entry
3) Update entry
4) Delete entry
5) Show entry details
6) Exit
Enter choice: 1

All timetable entries:
[95bfb730] Friday 2:00-3:00 | calculus | Mani maran | 341
```

UPDATED ENTRY

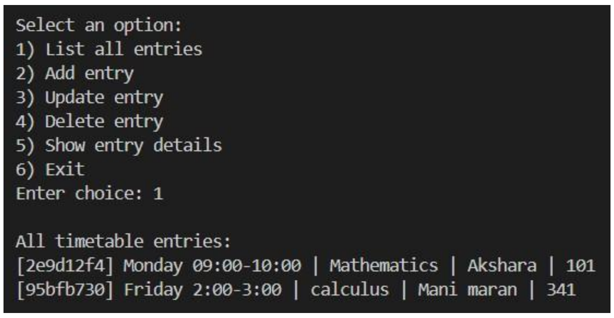


```
Select an option:
1) List all entries
2) Add entry
3) Update entry
4) Delete entry
5) Show entry details
6) Exit
Enter choice: 3
- 2e9d12f4 | Monday 09:00-10:00 | Mathematics | Akshara | 101
- 95bfb730 | Friday 2:00-3:00 | calculus | Mani maran | 341
Enter entry ID: 2e9d12f4
Press Enter to keep current value.
Subject [Mathematics]: Algebra
Teacher [Akshara]: Yash
Room [101]:
Day [Monday]:
Start time [09:00]:
End time [10:00]:
Notes [Calculus]:
Updated entry 2e9d12f4

Select an option:
1) List all entries
2) Add entry
3) Update entry
4) Delete entry
5) Show entry details
6) Exit
Enter choice: 1

All timetable entries:
[2e9d12f4] Monday 09:00-10:00 | Algebra | Yash | 101
[95bfb730] Friday 2:00-3:00 | calculus | Mani maran | 341
```

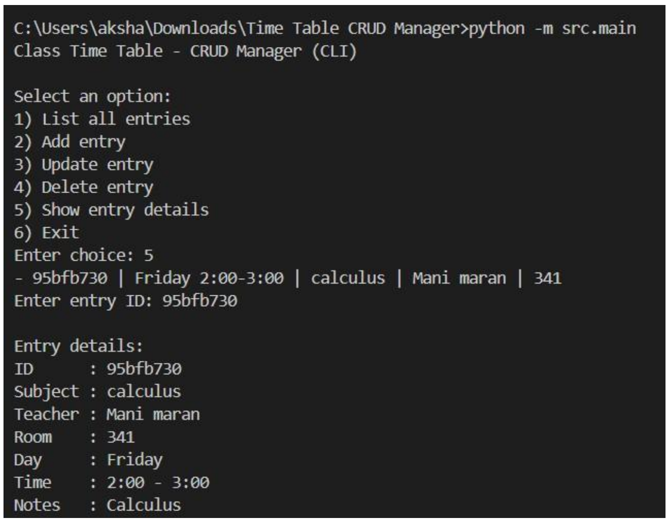
LIST ALL ENTRIES



```
Select an option:
1) List all entries
2) Add entry
3) Update entry
4) Delete entry
5) Show entry details
6) Exit
Enter choice: 1

All timetable entries:
[2e9d12f4] Monday 09:00-10:00 | Mathematics | Akshara | 101
[95bfb730] Friday 2:00-3:00 | calculus | Mani maran | 341
```

ENTRY DETAILS



```
C:\Users\aksha\Downloads\Time Table CRUD Manager>python -m src.main
Class Time Table - CRUD Manager (CLI)

Select an option:
1) List all entries
2) Add entry
3) Update entry
4) Delete entry
5) Show entry details
6) Exit
Enter choice: 5
- 95bfb730 | Friday 2:00-3:00 | calculus | Mani maran | 341
Enter entry ID: 95bfb730

Entry details:
ID      : 95bfb730
Subject : calculus
Teacher : Mani maran
Room    : 341
Day     : Friday
Time    : 2:00 - 3:00
Notes   : Calculus
```

9.Testing Approach

The testing approach for the Timetable CRUD Manager mainly involved manual, scenario-based testing. Each feature—Add, View, Update, Delete, and List—was tested individually to ensure the system responded correctly to user actions. Valid inputs were tested to confirm that operations worked smoothly, while invalid inputs were used to check error handling, such as entering wrong menu numbers, missing data, or incorrect time/date formats. Multiple entries were added to test consistency over repeated operations, and the file storage was reviewed after each change to ensure the data was saved accurately. Regression testing was performed whenever a change was made to confirm no previous functionality broke.

10.Challenges Faced

Some challenges included designing a simple yet effective menu layout, ensuring smooth navigation, and preventing the program from crashing when users entered invalid inputs. Handling file storage without causing data duplication or corruption required careful attention. Structuring the system into layers (UI, logic, storage) also took thoughtful planning to keep the code clean and understandable.

11. Learnings & Key Takeaways

The project enhanced understanding of CLI design, CRUD operations, persistent data storage, and code organization. It improved debugging skills and highlighted the importance of validation and error handling. The project also demonstrated how small, well-structured programs can still provide useful and practical functionality.

12. Future Enhancements

Future improvements could include adding a graphical user interface (GUI), integrating a database, adding search and filter features, enabling reminders or notifications, and supporting export/import of timetable data.

13. References

Programming basics, CRUD operation concepts, file-handling documentation, and general CLI design guidelines.