****

**PURBANCHAL UNIVERSITY**

KULESHWOR AWAS CAMPUS

FIRST SEMESTER PROJECT

ON

“CALENDAR MANAGEMENT SYSTEM”

In the partial fulfillments for the requirement of the first Semester Project-I (Subject code-………) in the completion of Bachelor of Information Technology (BIT) degree at Kuleshwor Awas Campus, under Purbanchal University.

Submitted By: Submitted To:

Name: Pasang Dorje Tamang Purbanchal University

S.N: 312113

Name: Prasanna Katuwal

S.N: 312115

2025-6-9

# ACKNOWLEDGEMENT

We take this opportunity to express our profound appreciation and unfathomable regards to the Information Technology (IT) department for this commendable guidance, monitoring and constant encouragement throughout the course of this project. The help and guidance given by shall carry us a long way, in the journey on which we are about to embark.

We also take this opportunity to express a deep sense of gratefulness to Mr. Narayan GC for his amiable support, valuable information and guidance, which helped us in completing this task throughout its various stages. We also want to thank our program director Mr. Narayan G.C. who helped us a lot during the complete project by giving us his precious time. We are also indebted to all members of Kuleshwor Awas Campus, for the help provided by them in their in their respective fields. We are grateful for their cooperation during the period of our project.

Finally we would also like to express lots of thanks to PURBANCHAL UNIVERSITY for designing such a wonderful course structure. It will help us to get more knowledge in the field of Information Technology & help us to have a bright future in the field of technology.

# STUDENT’S DECLARATION

We following students, hereby declare that the Project Report is titled

“Calendar Management System” is a result of our own work and our indebtedness to other work publications, references, if any, have been dully acknowledged. If we are found guilty of copying any other report or published information and showing as our original work, we understand that we shall be liable and punishable by Purbanchal University, which may include fail in examination, ‘Repeat study and re-submission of the report’ or any other punishment the Purbanchal University may decide.

We further certify that this Project submitted in partial fulfillment of the requirement for the award of Bachelor in Information Technology (BIT) of the Purbanchal University is our original work and has not been submitted for award of any other degree or other similar title or prizes.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.N | Name | Redg | Roll no. | Signature |
| 1. | Pasang Dorje Tamang | 177-3-2-01980-2024 | 312113 |  |
| 2. | Prasanna Katuwal |  |  |  |

# 

# EXAMINER’S CERTIFICATION

This is to certify that the project entitled “CALENDAR MANAGEMENT SYSTEM” has been successfully completed by Mr. Pasang Dorje Tamang (S/N: 312113) & Mr. Prasanna Katuwal (S/N:312115) in partial fulfillment of Degree of Bachelor of Information Technology of Purbanchal University during the academic year 2016

Under the guidance of Mr. Narayan G.C.

………………………………………………….

Department of Science and Technology

Kuleshwor Awas Campus

PURBANCHAL UNIVERSITY

**----College Letterhead**

#### **To Whom It May Concern**

This is to clarify that Mr. Pasang Dorje Tamang and Mr.Prasanna Katuwal, students of Bachelor in information technology (BIT) program, have successfully completed their first-semester project named “Calendar Management System”.

This project is the original work of Mr.Pasang Dorje Tamang and Mr.Prasanna Katuwal, carried out under the supervision of Mr. Narayan G.C. as per guideline provided by Purbanchal University, and is certified as per the student’s declaration that the project “Calendar Management System” has not been submitted or presented

Elsewhere as part of any academic work.

The details of the student are as follows:-

Name of student: -Pasang Dorje Tamang, Prasanna Katuwal

Course Semester: - first Semester

Subject: - Project-I

Subject Code: - BIT106CO

……………………………………..

Narayan G.C.

Program Director, BIT

KULESHWOR AWAS CAMPUS

# ABSTRACT

The Calendar Management System is a console-based application developed in the C programming language that facilitates efficient personal scheduling and event management. This system enables users to view monthly calendars, manage events, and maintain a to-do list. It offers core functionalities such as viewing a calendar for a specified month and year, adding and deleting events, and displaying stored events in a readable format. Additionally, the integrated to-do list module allows users to add tasks, mark them as completed, and review pending or completed tasks.

The calendar view takes leap years into account and computes the correct day of the week for the beginning of each month, ensuring accurate calendar representation. Events are stored in a text file (event.txt), making the system simple yet persistent across sessions. The to-do list operates similarly, storing active tasks in todo.txt and completed tasks in complete.txt.

Designed with simplicity and utility in mind, the Calendar Management System offers a practical solution for students and professionals to organize their time and activities without needing complex software. Its file-based architecture ensures data persistence while maintaining a lightweight footprint. This project demonstrates fundamental programming concepts such as file handling, modular function design, and user input validation, and basic date computations.

Table of Contents

[STUDENT’S DECLARATION I](#_Toc201163349)

[II](#_Toc201163350)

[EXAMINER’S CERTIFICATION II](#_Toc201163351)

[ACKNOWLEDGEMENT III](#_Toc201163352)

[ABSTRACT IV](#_Toc201163353)

[List of Figures VI](#_Toc201163354)

[1. BACKGROUND 1](#_Toc201163355)

[1.1 INTRODUCTION 1](#_Toc201163356)

[1.2 OBJECTIVE OF THIS PROJECT 1](#_Toc201163357)

[1.3 FUTURE IMPLEMENTATION FOR THE PROJECT 2](#_Toc201163358)

[1.4 PROCESS IN THE PROJECT USE 3](#_Toc201163359)

[1.5 INTRODUCTION TO SOFTWARE / LANGUAGE USED 3](#_Toc201163360)

[2. SYSTEM RECOMMENDATIONS: 4](#_Toc201163361)

[3. PROBLEM FACED 4](#_Toc201163362)

[4. System design 5](#_Toc201163363)

[Step 1: Display main menu with options: 5](#_Toc201163364)

[Step 2: Wait for user input (1–6) and process based on selected option. 5](#_Toc201163365)

[Step 3: **If Option 1 (View Calendar):** 5](#_Toc201163366)

[Step 4: If Option 2 (Add Event): 5](#_Toc201163367)

[Step 5: If Option 3 (View Event): 5](#_Toc201163368)

[Step 6: If Option 4 (Delete Event): 6](#_Toc201163369)

[Step 7: If Option 5 (To-Do List): Display submenu: 6](#_Toc201163370)

[Step 8: If Option 6 (Exit): 6](#_Toc201163371)

[Step 9: Stop 6](#_Toc201163372)

[5. SNAPSHOTS OF THE SYSTEM 8](#_Toc201163373)

[6. LIMITATIONS 10](#_Toc201163374)

[7. CONCLUSIONS 11](#_Toc201163375)

[8. BIBLIOGRAPHY 12](#_Toc201163376)

# List of Figures

[Figure 1 Flow Chart 8](file:///C:\Users\Sptamang\Desktop\Project%20Report.docx#_Toc201162467)

[Figure 2 Main Menu 9](file:///C:\Users\Sptamang\Desktop\Project%20Report.docx#_Toc201162468)

[Figure 3 Calendar 9](#_Toc201162469)

[Figure 4 Can;t add event in Past Dates 9](#_Toc201162470)

[Figure 5 To-do list menu 9](file:///C:\Users\Sptamang\Desktop\Project%20Report.docx#_Toc201162471)

[Figure 6 View To-do list 10](file:///C:\Users\Sptamang\Desktop\Project%20Report.docx#_Toc201162472)

[Figure 7 Mark To-do as complete 10](file:///C:\Users\Sptamang\Desktop\Project%20Report.docx#_Toc201162473)

[Figure 8 Completed TO-DO 10](file:///C:\Users\Sptamang\Desktop\Project%20Report.docx#_Toc201162474)

# Chapter 1: Introduction

## INTRODUCTION

Time management plays a crucial role in both personal and professional life. As our daily tasks, responsibilities, and deadlines continue to grow, having a well-organized scheduling system has become more important than ever. A Calendar Management System helps users keep track of important dates, plan events, and manage tasks efficiently.

Traditionally, people rely on notebooks or basic reminders, which can be unreliable and hard to maintain over time. Our project, "Calendar Management System", aims to provide a simple yet effective digital solution to these problems. It allows users to view monthly calendars, add and manage events, and maintain a to-do list with the ability to mark tasks as completed.

This system is especially helpful for students, professionals, and anyone looking to organize their daily life more efficiently without relying on external applications. It reduces the chances of missing important dates and helps users stay productive and on schedule.

## OBJECTIVE OF THIS PROJECT

* To provide a simple and efficient way for users to manage dates, events, and daily tasks.
* To help users schedule and view events on specific dates with ease.
* To improve time management by allowing users to maintain and track a personal to-do list.
* To reduce the chances of missing important dates or tasks.
* To minimize the use of paper for planning and organizing schedules.
* To ensure faster access to past and upcoming events and tasks through a user-friendly interface.
* To avoid data redundancy by storing events and tasks in organized digital format using file handling.

## **METHODOLOGY**

To build this Calendar and To-Do List Management System, we followed a step-by-step approach that allowed us to develop the application in a structured and manageable way. The project was built using the C programming language and was designed to run as a console application for simplicity and easy interaction.

Step 1: Planning and Feature Selection

The first step was to clearly outline what features we wanted our system to have. We decided that the program should:

* Display a calendar for any given month and year
* Allow users to add and view events on specific dates
* Let users manage a simple to-do list, with options to add, view, complete, and review tasks

We kept the scope manageable while still ensuring useful real-world functionality.

### Step 2: Designing the User Interface

Since this is a terminal-based program, we focused on creating a menu-driven interface. The user is always presented with clear options (e.g., view calendar, add event, manage to-do list), and after completing each task, they’re brought back to the main menu. This design keeps navigation straightforward and user-friendly.

### Step 3: Building the Calendar Logic

We began the coding process by first implementing the calendar view feature. This required determining the correct weekday for the first of each month, accounting for leap years, and printing the calendar in a readable format. We wrote functions like checklp() to check for leap years and calcTotalDays() to calculate the number of days since year 0 to determine the starting weekday.

### Step 4: Implementing Event Management

Once the calendar worked correctly, we added the ability to manage events. We allowed users to:

* Add events for future dates
* View all saved events
* Delete events by specifying the date

We used file handling (fopen, fprintf, fgets, etc.) to store and retrieve these events from a file named event.txt, so they wouldn't be lost when the program closed.

### Step 5: Developing the To-Do List

Next, we focused on the to-do list feature. This part included the ability to:

* Add new tasks
* View current tasks
* Mark tasks as completed (moving them to a separate file)
* View completed tasks

We used separate text files (todo.txt and complete.txt) to keep track of pending and finished tasks. When a task was marked as complete, it was moved from one file to another.

### Step 6: Testing and Validation

After each major feature was implemented, we tested it to ensure it worked as expected. We also handled edge cases, such as trying to add events in the past, entering empty to-do items, or trying to mark a non-existent task as complete. We refined the program based on the results of these tests.

### Step 7: Final Touches

We added formatting to improve the output readability and made sure all files are properly opened and closed to avoid memory leaks or file corruption. We also ensured the program gracefully handles invalid input and guides the user back to the correct operation.

By following this methodical approach, we were able to build a fully functioning and user-friendly calendar and task manager. The experience also deepened our understanding of core C programming concepts, especially file handling, control structures, and modular design.

## SOFTWARE / LANGUAGE USED

C is a powerful general-purpose programming language developed in the early 1970s by Dennis Ritchie at Bell Labs. It is widely used for system programming and application development due to its efficiency and ability to work closely with hardware. C is considered a middle-level language because it combines features of both high-level and low-level programming. Its simplicity and speed make it well-suited for developing software like this calendar management system, which involves handling dates, events, and task management efficiently.

## TIMELINE

The project was developed over a period of 2 to 3 months and was divided into clear, structured phases to ensure smooth progress. The timeline outlines each stage of development, starting from idea selection and requirement analysis, moving through coding and implementation of features like the calendar, event manager, and to-do list, and ending with testing, documentation, and final submission. This organized progression helped maintain focus, track milestones, and ensure that each module was completed effectively before moving on to the next.

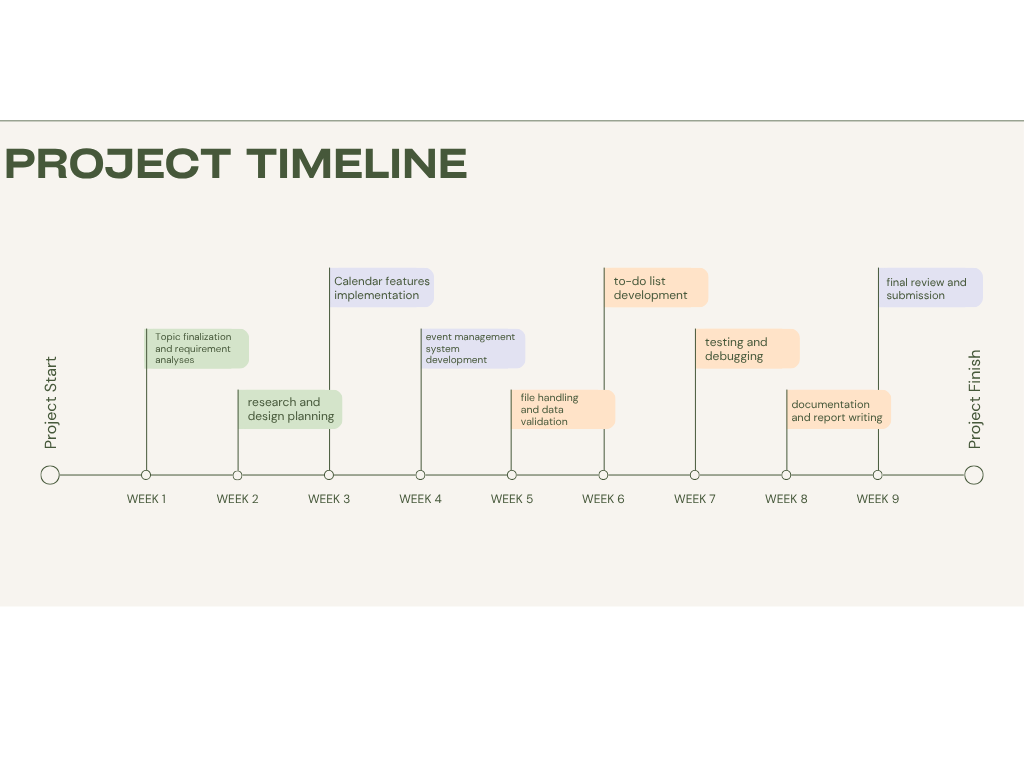


Figure 1.5.1 Timeline

# Chapter 2: SYSTEM REQUIREMENT

2.1 HARDWARE

To ensure smooth development and execution of the project, the following hardware components were required and used during the implementation phase.

* PC with Pentium II Processor (260 MHz) or higher
* 32 MB RAM or more
* Color Monitor (preferred)
* Hard disk with at least 50 MB of free space

2.2 SOFTWARE

The successful development and execution of the project relied on specific software tools and environments that supported coding, compilation, and testing of the system.

* The project requires a C compiler such as GCC or Turbo C to compile and execute the source code.
* An Integrated Development Environment (IDE) like Code::Blocks, Dev-C++, or Visual Studio Code can be used to write and manage the code more efficiently.
* The system is compatible with Windows, Linux, or macOS operating systems.
* A text editor is needed to view and modify output files such as event.txt and todo.txt.

# Chapter 3: SYSTEM DESIGN

## 3.1 INTRODUCTION TO THE PROJECT

This project is a Calendar and To-Do List Management System developed in the C programming language. It provides users with a simple and efficient way to view monthly calendars, manage events by adding, viewing, and deleting them, and organize daily tasks through a to-do list with options to mark tasks as completed. The system ensures data persistence using file handling and offers a user-friendly menu-driven interface for easy navigation.

## 3.2 ALGORITHM

The algorithm for the Calendar and To-Do List Management System is designed to handle date calculations, event management, and task tracking efficiently. It starts by calculating the total number of days to determine the day of the week for any given date, taking leap years into account. For event management, the system allows users to add, view, and delete events while ensuring that past dates cannot be selected. The to-do list module supports adding tasks, viewing pending and completed tasks, and marking tasks as complete. File handling is used throughout to store and retrieve data, ensuring persistence across sessions. The algorithm is implemented through modular functions to maintain clarity and ease of debugging.

The system consist of different sets of algorithms:

### **Step 1:** Display main menu with options:

* View Calendar
* Add Event
* View Event
* Delete Event
* To-Do List
* Exit Program

### **Step 2:** Wait for user input (1–6) and process based on selected option.

### **Step 3: If Option 1 (View Calendar):**

* Ask for year and month.
* Calculate first weekday of the month.
* Display calendar for selected month.

**Step 4: If Option 2 (Add Event):**

* Ask for event date.
* Validate that it’s a future date.
* Ask for event description.
* Save to event.txt.

### **Step 5: If Option 3 (View Event):**

* Read and display all events from event.txt.
* If none, show “No event found”.

### **Step 6: If Option 4 (Delete Event):**

* Show existing events.
* Ask for date to delete.
* Remove matching entry from file.

### **Step 7: If Option 5 (To-Do List):** Display submenu:

1. Add To-Do
2. View To-Do
3. Mark as Completed
4. View Completed
5. Return to Main Menu

Each sub-option performs corresponding file operations (todo.txt, complete.txt).

### **Step 8: If Option 6 (Exit):**

* Terminate program using return 0.

### **Step 9: Stop**

## 3.3 FLOWCHART

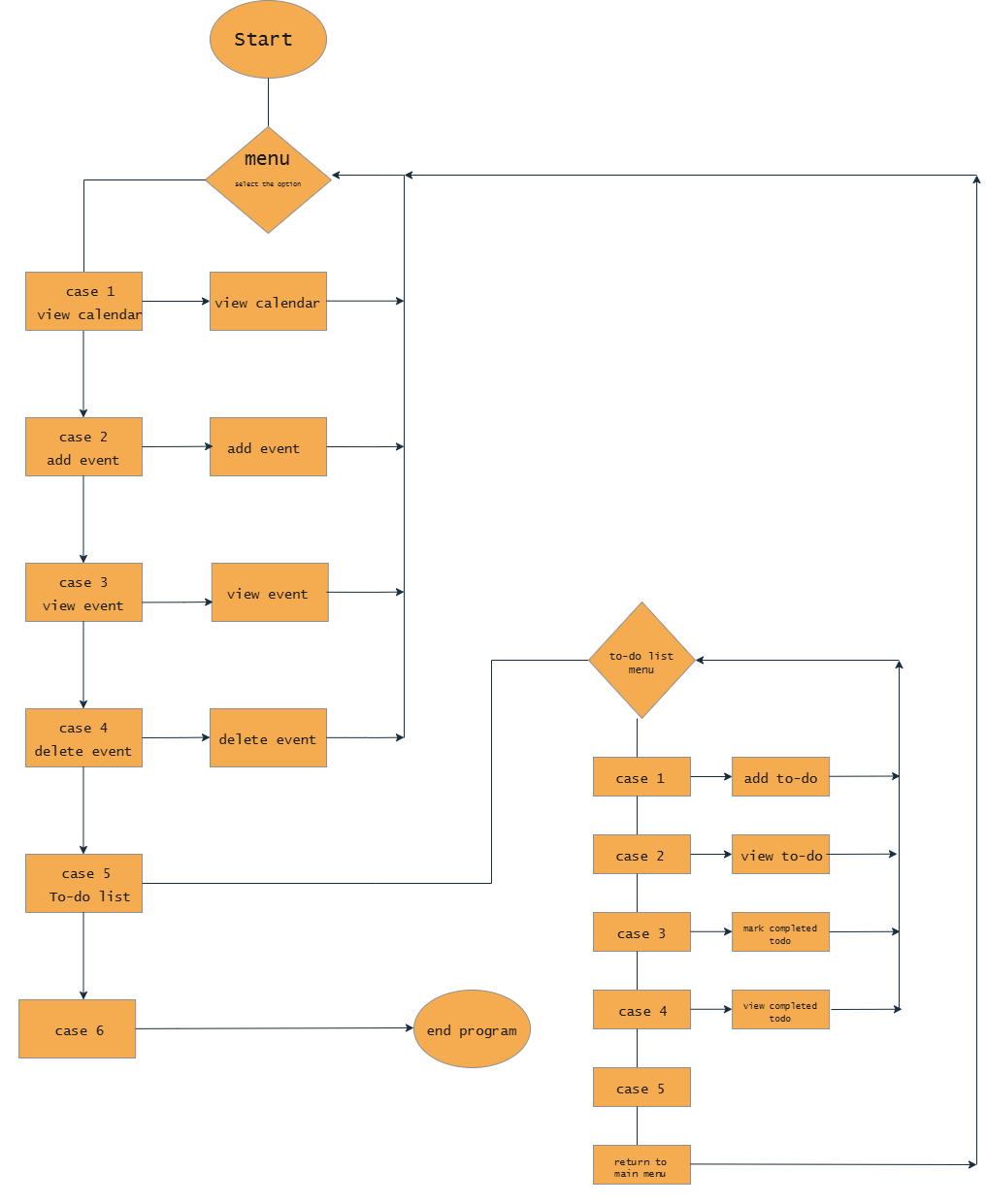
The flowchart represents the logical structure and sequence of operations in the Calendar and To-Do List Management System. It begins with the main menu, allowing users to choose between viewing the calendar, managing events, or accessing the to-do list. Each option leads to a specific set of operations such as adding, viewing, or deleting data. Decision-making blocks guide the program flow based on user input, while arrows indicate the transition between different processes. The flowchart helps visualize the overall functionality and makes it easier to understand the system's working.

Figure 3.3.1 Flow Chart

## 3.4 SYSTEM REQUIREMENT

## 3.4.1 FUNCTIONAL REQUIREMENT

3.4.1.1 MENU

The system provides a user-friendly, menu-driven interface that allows users to navigate between key functionalities such as viewing the calendar, managing events, and handling to-do lists. Each menu option leads to a specific set of features designed for ease of access and interaction.

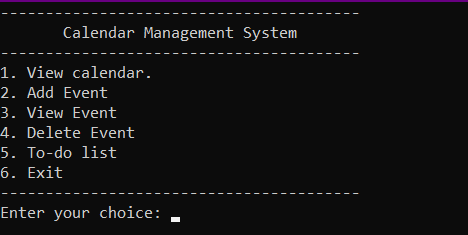


Figure 2.4.1.1 Main Menu

3.4.1.2 VIEW CALENDAR

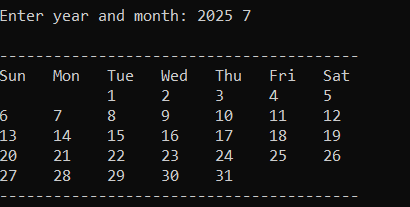
The view calendar feature allows the user to display the calendar of any specified month and year. It accurately shows the layout of days, taking into account leap years and correct weekday alignment.

Figure 3.4.1.2 view calendar

3.4.1.3 DATE VALIDATION

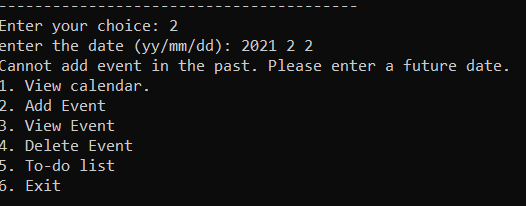
The system includes date validation to ensure that users cannot add events to past dates. It checks the entered date against the current system date to maintain logical consistency and prevent scheduling errors.

Figure 3.4.1.3 DATE VALIDATION

3.4.1.4 TO-DO LIST MENU

The to-do list menu provides options to add new tasks, view pending tasks, mark tasks as completed, and view completed tasks. It helps users manage daily activities efficiently within a simple and interactive interface.

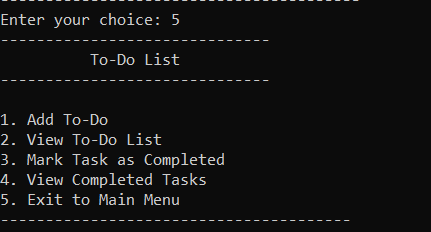


Figure 3.4.1.4 To-do list menu

3.4.1.5 VIEWING TO-DO LIST

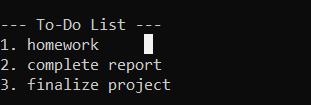
The viewing to-do list feature displays all pending tasks saved by the user. It reads data from the storage file and presents the tasks in a numbered list for easy tracking and management.

Figure 3.4.1.5 View To-do list

3.4.1.6 MARKING TO-DO AS COMPLETE

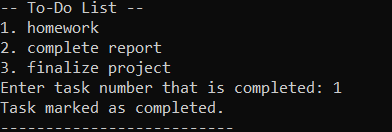


Figure 3.4.1.6 Mark To-do as complete

3.4.1.7 COMPLETED TO-DO

The completed to-do feature displays all tasks that have been marked as finished. It retrieves and lists these tasks from a separate file, allowing users to review their completed work and monitor productivity.

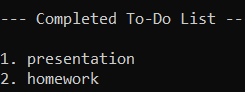


Figure 3.4.1.7 Completed TO-DO

3.4.1 NON-FUNCTIONAL REQUIREMENT

* The system should have a user-friendly, menu-driven interface for easy navigation.
* It must operate reliably without crashes and handle invalid inputs gracefully.
* The system should perform efficiently, ensuring quick response during calendar and task operations.
* It should be portable across different operating systems like Windows and Linux with minimal changes.
* The code should be modular and organized to allow easy maintenance and future updates.
* User data must persist between sessions through effective file handling and storage.

# CHAPTER 5: CONCLUSION

# LIMITATIONS

The following are the Limitations of the project-

* No graphical user interface; only runs in command-line mode.
* No reminder pop-up or event notification system.
* Does not validate invalid dates like 31/02/2025.
* No option to search or filter events.
* No login, password protection, or data security.
* Single-user only; does not support multiple user profiles.
* To-do tasks lack deadlines, priorities, and categories.
* Calendar view does not highlight event dates.
* Uses plain text files with no backup or recovery system.
* Not accessible via mobile or web platforms.

# CONCLUSIONS

Working on this Calendar Management System project has been a great learning experience for us. It gave us the chance to explore different concepts in C programming, especially related to file handling, modular functions, and date management. We really enjoyed building something that’s practical and useful, even if it runs in a simple command-line interface.

With the support of our teachers and friends, we were able to complete the project successfully. It wasn’t just about writing code — we also learned how to structure a complete program, handle errors, and think about how users will interact with it. These lessons will definitely help us in future academic projects and professional work.

At the same time, we understand that the project isn’t perfect. There are still many features we’d like to add — like reminders, a proper user interface, and maybe even online access. But as a starting point, this project helped us build confidence and gave us a strong foundation to work on more complex applications in the future.

# BIBLIOGRAPHY

* sscanf for file label. <https://www.google.com/search?q=sscanf+in+c+for+label+in+file+handling>.
* regEX validation.<https://stackoverflow.com/questions/1085083/how-to-use-regular-expressions-in-c>
* For time.<https://www.google.com/search?q=time+in+c>. youtube:<https://youtu.be/i1MeXMciy6Q?si=QWvUVshjlIkVIJT8>.
* Learning about new functions and syntax. <https://chatgpt.com> .
* Project supervisor Mr. Narayan G.C. for help and guidance.

.