ABSTRACT

The aim of our project is to create a console-based application developed in C that performs basic banking operations. The user can perform operations such as account creation, viewing account details, depositing and withdrawing balance, transferring balance among other users, and deleting an existing account.

On running the program, users will be asked to authenticate themselves. This can be done by either signing up for a new account or logging in to an existing account. After the user is successfully logged in to the system, the user will be provided a menu-based interface offering a range of options for the user to select. Users can perform the above-mentioned operations by selecting the options on the menu.

This project optimizes basic programming concepts such as user input validation, file handling, and menu-driven interfaces for a learner to create a basic banking system. Through this project, a learner can gain certain experience in software development.

Keywords: Authentication, File handling, Input validation, Menu-driven program

1. INTRODUCTION

1.1. Background Information

This project will solely be created using the C Programming Language. C provides us with many features such as file handling, control structures, arrays and such that will be the building block of our project. Our project asks users to log into the system after which the user can proceed with the task provided on the menu-based interface. We aim to make the interface as neat and user-friendly as possible.

1.2. Motivation

When we were provided with this task, we were looking for an interesting project we could do in C. We also wanted to do our project on something practical and something we came across regularly. All our members came together and after sharing what everyone had in their mind, we finally settled on the Bank management system.

Banks play an irreplaceable role in all of our life. The modern banking systems and the likes of ATMs have made our lives extremely comfortable. We can deposit, withdraw, and transfer funds from anywhere at any time we want. When we started thinking if we could make a simple version of a banking system using C, it became more and more doable. We believe we can successfully make a proper banking system which will provide us with some amount of confidence and experience for our future projects.

1.3. Objective

Our project aims to fulfill the following objectives:

- To understand the core concept of file handling, control structure, arrays, and structures in C and make optimum use of them.
- To design a user-friendly menu-based interface that works properly with no bugs and errors.

1.4. Scope and Application

We aim to create a banking system where users can deposit, withdraw, and transfer funds through a menu-based interface. Like any other banking system, we will have users set up their account with a username and password for themselves so they can only log in to their account and their money and details remain secured in the database.

With this banking concept, a user can safely deposit his funds in his bank account and retrieve it back at any moment. This will simplify their lives by cutting off the hassle of going to the bank for every simple task. Users can go to the nearest ATM where they can have access to withdraw, deposit, or transfer cash into other users' accounts.

4. METHODOLOGY

Our main task in this project was to gather various inputs from users and sort those data properly in our local storage so that we could fetch it again in various parts of the program. For that, the tools in C that we used include header files, various functions (both predefined and user-defined), control structures, arrays and structures, and file handling.

4.1. Header Files

A C header file is a text file that contains pieces of code written in the C programming language. The name of a header file, by convention, ends with the . h extension. It is inserted inside a program by coding the #include preprocessor directive. Each of the header files lets us use certain functions associated with them.

The header files we used in this project are:

- stdio.h
- windows.h
- stdlib.h
- string.h
- time.h

4.2. Functions

Function plays a pivotal role in any project. A function is a block of code that only runs when it is called. They are of two types: Pre-defined functions and user-defined functions.

Pre-defined functions are functions that are pre-built in C and we can simply import them through header files. On the other hand, User-defined functions are the functions that users can create themselves, which may take some arguments and may display or return an appropriate result.

In this project, we used various pre-defined functions as well as created various functions that would be commonly used again and again in our code. The use of function reduces the repetition of codes and makes our program more efficient and human-readable.

4.3. Control Structure

Control structures analyze certain variables and choose a direction in which the program is to be executed. They are fundamentally classified into Branching and Looping structures.

4.3.1. Branching

Branching statement controls the flow of execution of the program. Branching statement can be either Conditional or Unconditional.

Conditional Branching is when our program executes a block of statement only when a certain condition is met. Conditional Branching Statements include if statement, if-else statement, nested if-else statement, and switch statement.

Unconditional Branching is used when we want to jump into a certain line or part of the code regardless of any condition. Commonly used Unconditional Branching Statements include goto statement, break statement, and continue statement.

4.3.2. Looping

Looping fundamentally means repetition. In programming, looping includes a block of code that is repeated over and over until a certain condition is met. Looping is very useful in C to avoid repetition of codes. Looping can be Entry Controlled or Exit Controlled loop.

In Entry Controlled Loop, the test expression is checked at the beginning of the code. Only if the test condition is met true, the code is further executed. Once the code is done the test condition is checked again. This process repeats until the condition is met false.

In Exit Controlled Loop, the program first lets a block of code be executed and at the end of the block, a test expression is checked. Only if the test expression is false, our program continue. If not, the block of code is repeated again and again.

4.4. Arrays and Structure

Arrays are used to store multiple values of the same data types in a single variable, instead of declaring multiple variables. They are very useful when we have to ask the user to input a large amount of data of the same data types and we need to access it again in some part of the program.

Structures are used to store multiple variables of different data types. Different related variables which can be of different data types are grouped in one place with the help of structure. We can easily access these variables again in any part of the program.

In our project, we have used structures along with arrays to save every detail of a user like their username, password, balance, and statement history in a single structure.

4.5. File Handling

File Handling in C is the process in which we create, open, read, write, and close operations on a file. Most of the functions used in file handling are easily accessed by the header file stdio.h. These functions includes fopen(), fscanf(), fprintf(), fwrite(), fclose().

In our project, we have made use of file handling to take the details entered by the user and store them in a separate file so that even when the user closes the program, their details are saved and their record doesn’t get deleted.

5. TIME ESTIMATION

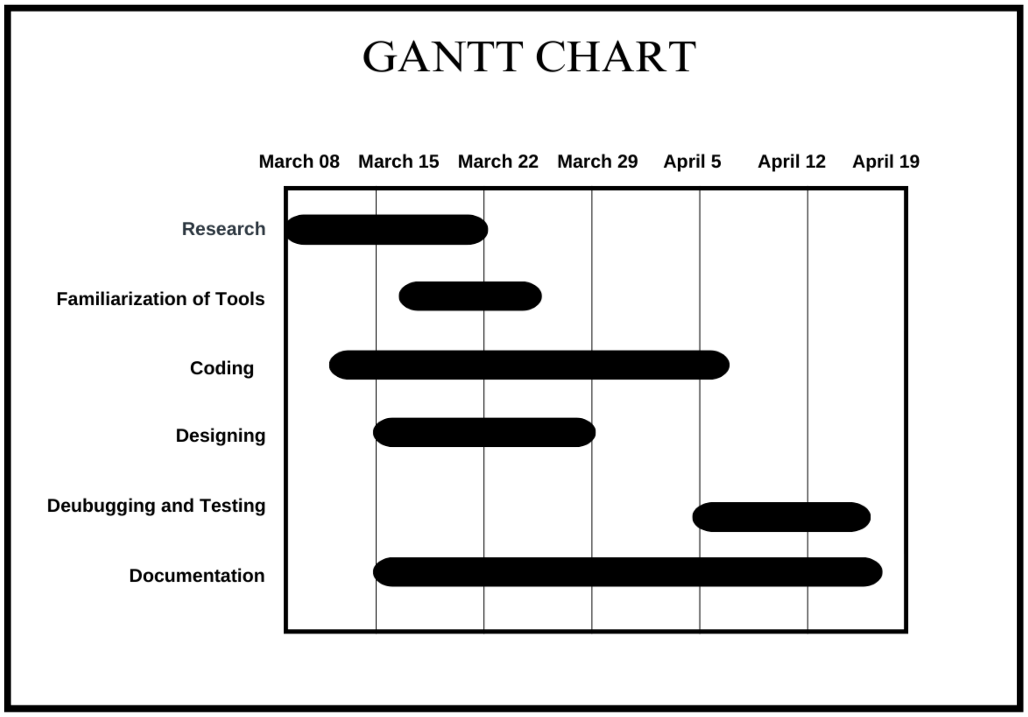

Table 5.1: Gantt Chart with Time Estimation

6. FEASIBILITY ANALYSIS

The main goal of feasibility analysis is to test if this project is worth doing. We will be analyzing the Economic, Technical, and Scheduling Feasibility.

6.1. Economic Feasibility

This project is built as a part of an academic project work and is not meant to be profitable. No additional hardware devices/software are needed to be bought either for building this project. Every member involved has their own computing devices which is sufficient for them to contribute to this project. So, no additional budget is allocated for this project.

6.2. Technical Feasibility

The technical feasibility of developing a banking system appears favorable based on our assessment. The complexity of our project is fairly low. Furthermore, the technology and tools necessary are well understood by our team members.

6.3. Scheduling Feasibility

The complexity of the program and the experience of our members suggest that our project is achievable in the given timeframe. Our project plan incorporates a realistic time frame for each phase of our project. We have made our project plan by taking our school work as well as the availability of our members into account. So, based on our comprehensive scheduling, we are confident in the feasibility of completing this project in the allocated timeframe.