TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

THAPATHALI CAMPUS

Proposal

On

C-Pay: An E-wallet Application Utilizing C-Programming

Submitted By:

Aswin Kandel (THA081BCT004)

Dikesh Manandhar (THA081BCT008)

Kishan Kumar Shah (THA081BCT014)

Pujag Dallakoti (THA081BCT024)

Submitted To:

Department of Electronics and Computer Engineering

Thapathali Campus

Kathmandu, Nepal

February 2025

ACKNOWLEDGEMENT

We sincerely appreciate the Institute of Engineering, Thapathali Campus, for providing us with this project as an opportunity to advance our skills and knowledge. We sincerely thank the Department of Electronics and Computer Engineering for their continuous support and for giving us access to vital data and resources. We especially thank Er. Prajwol Pakka and Er. Anup Shrestha, our supervisors, for their invaluable advice, assistance, and knowledgeable insights during this study. Their advice was essential to our project's successful completion.

We also want to express our appreciation to our seniors and colleagues for their encouragement and support. This project allowed us to apply our practical System Development skills, especially with the C programming language, and was more than just an academic assignment. The practical application and educational experience have been immensely beneficial, and we are grateful to everyone who helped us along the way.

Aswin Kandel (THA081BCT004)

Dikesh Manandhar (THA081BCT008)

Kishan Kumar Shah (THA081BCT014)

Pujag Dallakoti (THA081BCT024)

ABSTRACT

The rapid global shift to internet transactions emphasizes how urgently digital financial solutions are needed, especially in underdeveloped countries like Nepal where traditional banking practices are the norm. Our project offers a working prototype that will eliminate the need for face-to-face banking encounters by streamlining small, necessary transactions over a digital platform. We have developed a digital wallet that makes it simple for customers to make payments using the C programming language, which is well known for its robust system-level features. In addition to making financial transactions easier, this endeavor makes use of C's educational components, integrating all the ideas and strategies we studied throughout our first semester.

Keywords: C Programming, Digital Wallet, Online Transactions, Procedural Programming, System Development

Table of Content

List of Figures

List of Abbreviations

ANSI	American National Standards Institute
AT&T	American Telephone & Telegraph
BCT	Bachelor in Computer Technology
C11	C Standard Revision 2011
C99	C Standard Revision 1999
GCC	GNU Compiler Collection
GDB	GNU Debugger
IDE	Integrated Development Environment
OS	Operating System
UI	User Interface

1. INTRODUCTION

As part of the first semester of the Bachelor of Computer Technology (BCT), this project aims to develop a small yet important C-programming experiment. The goal is to use the fundamental ideas of the sophisticated programming language C to create a prototype online payment system known as C-pay. By allowing users to register, log in, and make direct payments to other users or organizations, this platform seeks to facilitate seamless financial transactions. This project highlights important features and serves as a hands-on implementation of the theoretical ideas learned throughout the course, providing a practical interface for our growing programming skills.

1.1 Background

As first-year Bachelor in Computer Technology (BCT) students, we begin learning C programming in our Computer Programming-I course to develop our logical reasoning and problem-solving abilities. Our goal for the semester project was to use basic C programming concepts like file handling, pointers, structures, and arrays to create a prototype for Nepal's first online payment gateway, appropriately named C-pay. Users can register, log in, and carry out financial activities including money transfers and bill payments using this system. Our approach involves breaking up the development process to effectively assign tasks to team members and making use of online collaboration tools to mitigate the challenges posed by distant learning to the greatest extent feasible. This project helps us put theoretical knowledge into practice while also preparing us to support Nepal's financial services' digital transformation.

1.2 Motivation

Our research was inspired by the revolutionary impact of digital wallets around the world and Nepal's growing reliance on these technologies. We chose to study system development using C programming because, as first-semester BCT students, we were captivated by the complexities and potential of systems like eSewa. This project, called C-pay, is our first effort to create a digital payment system. It's a thrilling opportunity for us to apply what we've learned in class to a practical setting, strengthening our understanding of programming and the realm of digital finance. To aid in our

development process, we have used a variety of tutorials and internet resources, which has given us a great opportunity to learn and put our theoretical knowledge into practice.

1.3 Problem Definition

During our first discussions about developing a digital wallet prototype, C-pay, we identified key issues that needed to be resolved to ensure the system's effectiveness and usability. The primary concerns include ensuring robust security to protect users' transactional and personal data, developing a logical system to handle account transfers and payments, and putting in place a safe user authentication process for signing in. These issues are crucial for preventing unwanted access and ensuring the wallet operates securely and consistently.

To promote usability and reduce user errors, it is also essential to design a user interface (UI) that is clear and easy to use. From registering to completing transactions, the interface should provide simple navigation and answers for each user's action. Furthermore, the technical feasibility of the project, given our current proficiency in C programming, calls for careful planning and time management to successfully integrate the contributions of every team member and meet our academic and project deadlines without compromising the output's quality and functionality.

1.4 Objective

The main objectives of our project are listed below:

- To develop a Prototype: Create functional e-wallet prototype using C programming to handle financial transactions.
- To enhance Skills and Collaboration: Improve understanding of C programming and teamwork through practical application and group collaboration.

1.5 Scope and Limitations

1.5.1 Scope & Applications

Our project is designed to simulate basic functionalities of a digital wallet system like Esewa. It includes features such as:

- Secure Data Storage: User data, including passwords, is stored in binary files with a .dat extension, enhancing security as the contents are not easily readable.
- Encryption: Passwords are protected using a circular encryption mechanism, requiring a secret cipher code for decryption.
- Transaction Capabilities: Users can transfer money to other users and pay school fees.

1.5.2 Limitations

- Operating System Compatibility: The program is incompatible with 16-bit operating systems.
- Limited Transaction Features: Only transactions between personal accounts and payments to colleges are supported.
- Educational Scope: Some functions used in the program have not been covered in our coursework, limiting our ability to fully understand all aspects of the implementation.

2. LITERATURE REVIEW

2.1 Background of Online Transaction Systems

Online transaction technologies, which offer unmatched accessibility and simplicity, have significantly transformed the banking industry. These technologies eliminate the need for actual cash by enabling users to conduct financial transactions online. Digital wallets like PayPal, Alipay, and Apple Pay have set the norm globally by offering robust, user-friendly systems that manage everything from simple transfers to complex commercial transactions. Services like eSewa and Khalti have gained popularity in Nepal because they provide comparable features tailored to local needs, enabling users to send money, pay bills, and top up services online. These systems demonstrate how technology may be successfully incorporated into routine financial processes, reducing the need for traditional banking procedures. [1]

2.2 Evolution of C Programming Language

Developed in 1972 by Dennis Ritchie at AT&T Bell Laboratories, the C programming language is fundamental to contemporary software development. C was first designed for system programming on the new Unix OS, but its straightforward syntax and reliable low-level memory access made it ideal for a wide range of applications. Over the years, C has influenced many other languages, including C++, which adds object-oriented features to improve its functionality. C's significance in the development of embedded systems, operating systems, and complex computing systems has been reaffirmed by its standardization as ANSI C and subsequent iterations like C99 and C11. Because of its adaptability to a wide range of hardware platforms, C is the ideal choice for applications like embedded systems and payment solutions that require direct hardware interaction. [2]

2.3 Applications and Importance of E-wallets

- Convenience: Enables users to conduct financial transactions anywhere, anytime.
- Speed: Transactions are completed in seconds.
- Reduced Cost: Lower transaction fees compared to traditional banking.
- Accessibility: Increases financial inclusion for unbanked and underbanked populations. [3]

2.4 Drawbacks and Limitations

- Security Risks: Vulnerable to hacking, phishing attacks, and other cybersecurity threats. [4]
- Dependence on Internet Connectivity: Requires consistent and stable internet access.
- User Interface Limitations: Terminal-based interfaces may not be as user-friendly as graphical interfaces.

2.5 Methodologies for Creating E-Wallet

Creating an e-wallet application in C involves several key programming and design considerations:

- Secure Data Handling: Utilize C's cryptographic libraries to encrypt and secure user data.
- Efficient Transaction Processing: Implement algorithms optimized for fast and reliable transaction handling.
- User Authentication: Design mechanisms for secure login, potentially integrating two-factor authentication to ensure user identity verification.
- Intuitive User Interface: Craft a user-friendly terminal interface using C's I/O functions, making navigation straightforward.
- Data Storage: Using file handling to store user data securely.

3. PROPOSED SYSTEM ARCHITECTURE

3.1 Proposed Block Diagram

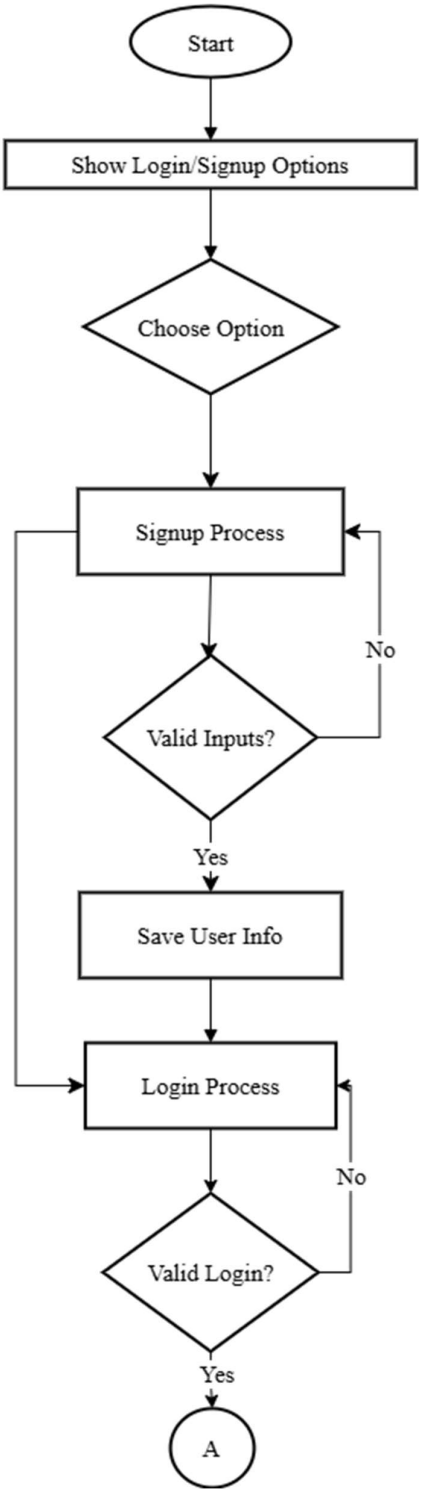


Figure 3-1-1: Flowchart for Main Program Flow

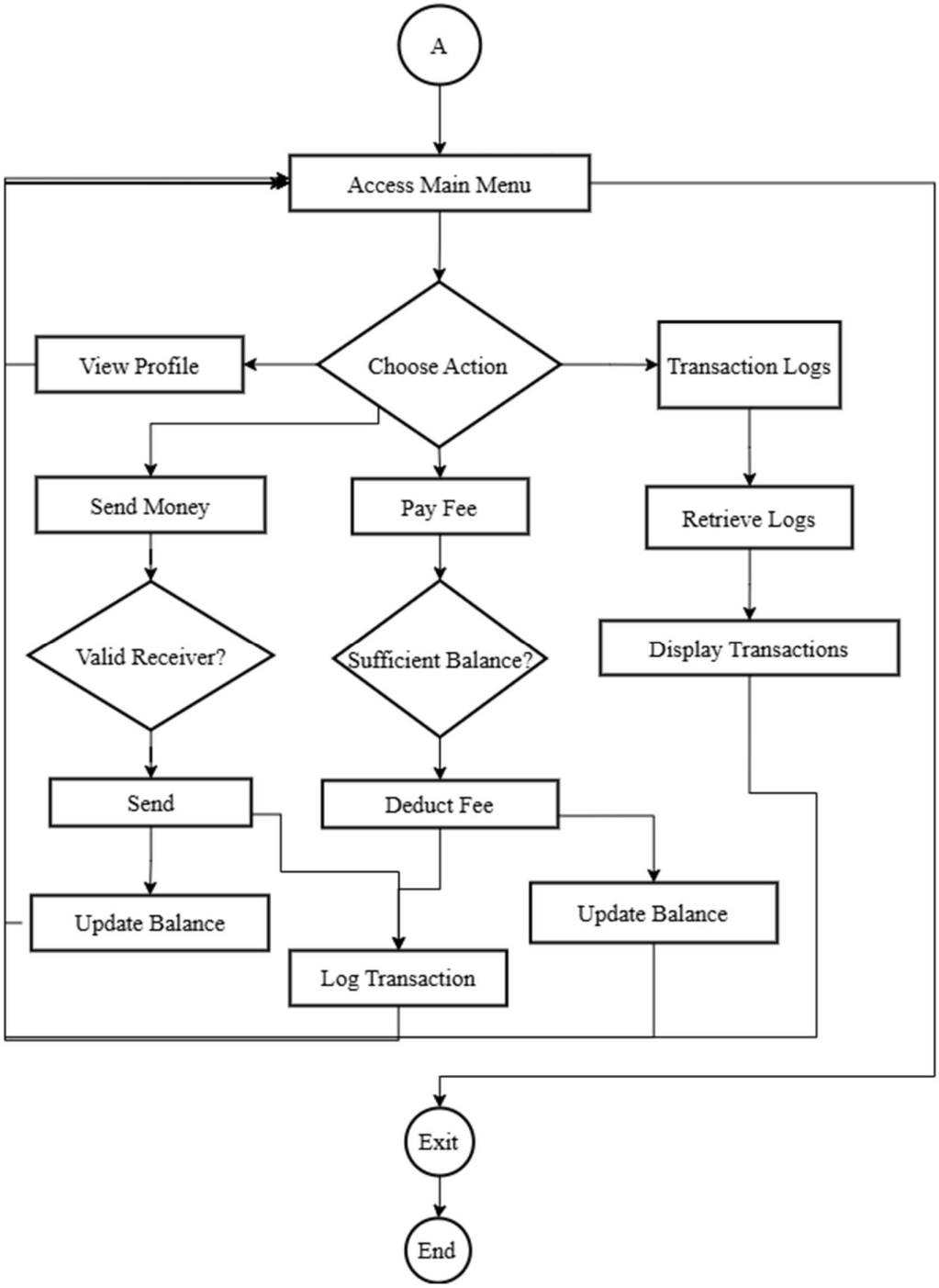


Fig 3-1-2: Menu Handling Flow

3.2 Program Flow

3.2.1 Main Program Flow

1. Start of Program:

   The application begins execution by invoking the cpay() function, which clears the screen and displays an ascii art.

2. User Interaction:

   Login or Sign Up:

   Users are prompted to either log in or sign up. The initial screen provides these two options.

3. Login Process (login() Function):

   Validates user credentials.

   If credentials are incorrect, the user is prompted to retry.

   Successful login transitions the user to the main menu.

4. Sign Up Process (signup() Function):

   Users must enter a username, email, phone number, and password.

   Input validation ensures:

   Usernames contain no space and are unique.

   Emails adhere to valid formats.

   Passwords are a minimum of four characters.

   Phone numbers are valid.

   Upon successful sign-up, user data is encrypted and saved, and the user is redirected to the login.

3.2.2 Main Menu Handling (menu_handling() Function):

1. Display Main Menu:

   Options are provided for checking balance and details, sending money, paying school fees, viewing transaction history, or exiting the program.

2. Option Execution:

Each menu option calls a specific function:

Check Balance and Details (showdetails()):

Retrieves and displays current user balance and personal details.

Send Money (send_money()):

Allows users to transfer funds to another user. Validates recipient details and updates balances accordingly.

Pay School Fee (pay_school_fee()):

Lists available of educational institutions and their associated fees. Validates funds availability and processes payments.

Transaction History (transaction_history()):

Displays a log of all user transactions with timestamps.

Exit Program (exit_program()):

Close the application with a farewell message.

3.2.3 Additional Functions

Password Handling:

password_taker(char[]):

Masks password input for security.

Data Encryption and Decryption:

encrypt(char[]) and decrypt(char[]):

Secure user data.

Utility Functions:

delay(int):

Implements a pause in execution to enhance user experience.

4. METHODOLOGY

The methodology for this project includes a systematic approach to developing and implementing C programming language-based user account management and transaction system. This part provides a clear understanding of how activities are carried out by outlining the sequence of tasks, tools, and techniques employed in the project.

4.1 User Interface Design

The user interface is the first point of interaction with the user. The interface is designed using ASCII art and simple command-line inputs to guide the user through login, sign-up, and transaction processes. Functions like cpay(), display_login(), and display_signup() are crucial for rendering the initial screens and facilitating user navigation.

4.1.1 Input Validation

All user inputs, such as username, password, and phone numbers, are validated through specific conditions in the signup() function to ensure data integrity and security. This includes checks for valid email formats, password strength, and phone number validity. [5]

4.2 Data Management

Data management involves handling user credentials and transaction details securely.

4.2.1 File Handling

User data is stored and retrieved using file operations in C. Functions such as login() and signup() involve reading from and writing to files, ensuring data persistence across sessions.[6]

4.2.2 Encryption and Decryption

Security is a critical component. The encrypt() and decrypt() functions are used to securely store user passwords. These functions modify the password characters by a fixed numerical value to obscure the original text. [7]

4.3 Transaction Handling

This subsystem is responsible for managing financial transactions between users.

4.3.1 Sending Money

The send_money() function allows users to transfer funds. It involves checking the recipient's details, validating the availability of funds, and updating balances accordingly.

4.3.2 Payment of Fees

The pay_school_fee() function handles payments to educational institutions listed within the system, deducting fees from the user's balance.

4.4 Tools and Environment

IDE: Code::Blocks or Visual Studio Code for writing and testing the C code.
Compiler: GCC to compile the code into an executable form.
Debugger: GDB for troubleshooting and debugging.

5. SCOPE AND APPLICATIONS

The project functions as a teaching aid by illustrating the principles of data management, user interaction, and system design in a console-based application. Users can investigate the following features with it:

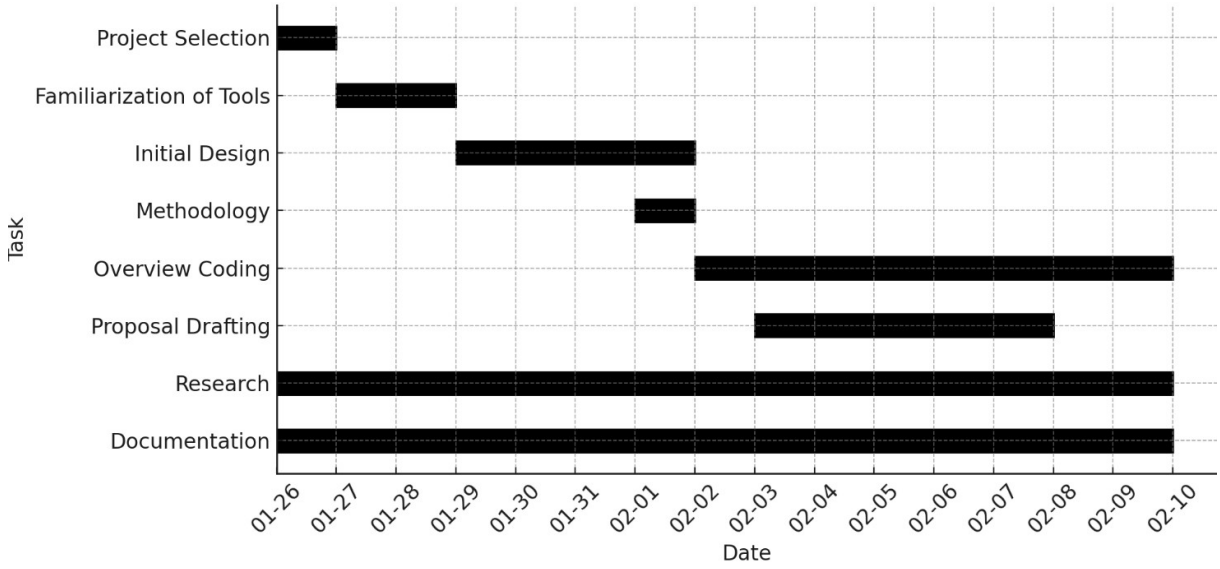
- **Educational Utility**: Students can understand how to use their theoretical understanding of algorithms, data structures, and programming in a real-world assignment. It provides the framework for more intricate systems.

- **Prototype Development**: Acts as a foundation for creating increasingly complex financial management systems with improved functionalities including real-time transaction updates and multi-factor authentication.

- **Customization for Various Uses**: Although the original configuration concentrates on personal transactions and college fees, it can be modified for small commercial purposes, such as handling transactions between departments within an institution or between vendors and customers.

6. ESTIMATED TIME SCHEDULE

Figure 6-1: Gantt Chart



7. FEASIBILITY ANALYSIS

This project's feasibility is assessed based on several criteria that ensure its practical implementation is both viable and educational.

- **Cost-Effectiveness:** The software development requires no financial investment in software licenses, as it utilizes open-source tools and environments. The primary cost involves human resources, primarily the time and effort of the student.

- **Technical Feasibility:** The use of C programming and basic file handling is appropriate for the skill level of a first-semester BCT student. These elements introduce students to fundamental programming concepts without the overhead of more complex languages or frameworks.

- **Resource Availability:** All necessary resources, such as compilers and development environments, are freely available. Documentation and community support for C programming are extensive, which aids in overcoming potential development challenges.

- **Scalability:** Initially, the project handles a small, predefined set of data and operations, but it can be scaled up to include more features such as different types of transactions, more detailed user profiles, and integration with actual banking APIs for real-time financial operations.

- **Maintainability:** The code is structured in modular functions, which makes it easier to maintain and update. Good programming practices and documentation are emphasized to ensure that the project can be easily understood and extended by other developers or as part of future coursework.

- **Educational Value:** The project provides practical experience with real-world applications of programming, enhancing the learning curve and offering a tangible outcome that reinforces theoretical knowledge.

References

[1] “ Digital Wallet ”, Wikipedia, Available:
https://en.wikipedia.org/wiki/Digital_wallet

[2] A. Devil, “ Programming-Basics ”, GitHub, Available:
https://github.com/Astrodevil/Programming-Basics

[3] “ Study on Digital Wallet Features ”, Kantar Public, Available:
https://www.ecb.europa.eu/press/pr/date/2023/html/ecb.pr230424_1_annex~93abdb80da.en.pdf

[4] “ E-Wallet Security Readiness ”, IJCSMC, Available:
https://ijcsmc.com/docs/papers/March2024/V13I3202410.pdf

[5] “ User Authentication in C Programming ”, LearnETutorials, Available:
https://learnetutorials.com/c-programming/programs/authentication-program

[6] “ File Handling in C ”, Stack Overflow, Available:
https://stackoverflow.com/questions/30788314/file-handling-in-c

[7] “ Encryption & Decryption Techniques ”, Stack Overflow, Available:
https://stackoverflow.com/questions/64922884/encrypting-decrypting-words-in-c