Project for 1st semester of Bachelor of Information Technology

**Invoice Hub**

****

**Ashbin Rai (310882)**

**Pratik Man Shrestha (310901)**

**Sujal Baidhya (310917)**

**KIST College of Information Technology**

**Faculty of Science and Technology**

**Purbanchal University, Nepal**

**February, 2025**

**Student’s Declaration**

We hereby declare that the project report entitled

**“Invoice Hub”**

Submitted in partial fulfillment of the requirements for the degree of Bachelor in Information Technology to Purbanchal University, Biratnagar, Nepal is our original work and not submitted for the award of any other degree, diploma, fellowship or any other similar title or prizes.

**Submitted by:**

Ashbin Rai (310882)

Pratik Man Shrestha (310901)

Sujal Baidhya (310917)

Date: 2025/02/14

**RECOMMENDATION**

This is to certify that this project entitled **Invoice Hub** **prepared and submitted by Ashbin Rai, Pratik Man Shrestha, and Sujal Baidhya,** in partial fulfillment of the degree of Bachelor of Information Technology awarded by Purbanchal University, and has been completed under my supervision. I recommend the same for acceptance by Purbanchal University.

Deepak Khadka, BIT Coordinator

Kist College of Information Technology

14 February, 2025

**CERTIFICATE**

This project entitled **Invoice Hub** prepared and submitted by **Ashbin Rai, Pratik Man Shrestha and Sujal Baidhya** has been examined by us and is accepted for the award of the marks of First Semester Project in BIT by Purbanchal University.

Name: Sailendra Basnet Signature Date: 2025/02/02

Associate Professor

Deepak Khadka Signature Date: 2025/02/02

Supervisor

Deepak Khadka Signature Date: 2025/02/02

Head of Department

**ACKNOWLEDGEMENT**

We are pleased to present the “**Invoice Hub**” as our 2nd semester project. We would like to express our sincere gratitude to all those who have supported and guided us throughout the duration of this project.

Firstly, we extend our deepest thanks to our **BIT Coordinator Mr. Deepak Khadka** whose expertise, guidance, and encouragement have been invaluable. His insights and feedback have greatly enriched this project. We are extremely grateful to our college, **KIST College of Information Technology** for giving to this opportunity and providing the resources and peaceful environment necessary for this project.

We are also grateful to our colleagues and team members, Ashbin Rai, Pratik Man Shrestha, Sujal Baidhya, for their active inspiration and cooperation. Our dedication and hard work have been instrumental in the successful completion of this project. We also thank **PURBANCHAL UNIVERSITY** for designing such a course structure, where we get to learn and implement new things

Lastly, we would like to thank our family and friends for their unwavering support and encouragement throughout this project.

**ABSTRACT**

This project, Invoice Hub, aims to streamline restaurant operations by automating the creation and management of customer invoices using C++ programming. The system features user authentication, menu management, and secure data handling, significantly enhancing billing accuracy and efficiency. It supports functionalities like updating prices and removing items, ensuring flexibility for restaurant staff. Future enhancements include cloud integration for data backup and remote access, UI improvements, and exploring IoT applications for further optimization. By reducing manual labor and minimizing billing errors, Invoice Hub demonstrates the practical application of C++ programming in solving real-world challenges in the hospitality industry. This project highlights the potential for further innovation in restaurant management systems.

**Keywords:** Invoice Generation, C++ Programming, Restaurant Operations, Automated Billing, Data Security, Future Enhancements.

Contents

[1. **Background** 8](#_Toc173480410)

[1.1 Introduction 8](#_Toc173480411)

[1.2 Objective of the project 8](#_Toc173480412)

[1.3 Features 9](#_Toc173480413)

[1.4 Future Prospects for Invoice Hub 9](#_Toc173480414)

[1.5 Programming Language Used 9](#_Toc173480415)

[2. **System Requirements** 11](#_Toc173480416)

[2.1 Hardware: 11](#_Toc173480417)

[2.2 Software: 11](#_Toc173480418)

[3. **System design** 12](#_Toc173480419)

[3.1 Algorithm 12](#_Toc173480420)

[3.2 Flowchart 19](#_Toc173480421)

[4. **Source Code** 22](#_Toc173480422)

[5. **System Analysis** 23](#_Toc173480423)

[6. **Problems Faced** 25](#_Toc173480424)

[7. **Conclusion** 26](#_Toc173480425)

[8. **References** 27](#_Toc173480426)

# 1. Background

## Introduction

Invoice Hub is a user-friendly invoicing system made for restaurants. It helps managers create, save, and manage invoices easily. With features like user sign-up, login, password recovery, and detailed invoice creation, Invoice Hub makes the billing process simple and accurate. This tool is designed to improve the efficiency of restaurant operations and enhance customer satisfaction. Additionally, Invoice Hub offers menu management, invoice search, and secure data handling, ensuring a comprehensive solution for all billing needs.

## 1.2 Objective of the project

The Primary Objective of this Project are as follows:

1. **Create Invoices Easily**: Simplify the process of generating bills for restaurant transactions.
2. **Handle Menu Updates**: Facilitate easy updates to the restaurant menu, including adding, updating, or removing items.
3. **Keep Data Safe**: Implement robust security measures to safeguard user and invoice data.
4. **Find Invoices Quickly**: Enable quick search and retrieval of past invoices for efficient record management.
5. **Easy to Use**: Design a system that's straightforward for restaurant staff to navigate and use.

## 1.3 Features

The major features of Invoice Hub are as following:

1. Create detailed bills for restaurant orders.
2. Easily manage menu items and their prices.
3. Search and display invoices of customer.
4. Securely handle user logins and data.
5. Simple interface with customization options.

## 1.4 Future Prospects for Invoice Hub

1. **Sales Analysis**: Get detailed reports on what sells best and when.
2. **Mobile App**: Make a version that works on phones and tablets
3. **Stock Control**: Help keep track of what's in stock and when to order more.
4. **Language Options**: Add more languages for different customers and staff.

## 1.5 Programming Language Used

C++ programming is an advanced language developed in the 1980s by Bjarne Stroustrup. It is known for its flexibility and support for both low-level and high-level programming. It builds upon C by adding object-oriented features like classes and inheritance, making it more versatile and scalable. C++ is widely used in system programming, game development, and applications requiring high performance due to its efficiency and control over system resources. Its strong emphasis on modularity and reusability has influenced many modern programming languages used today.

Main Concepts of C++ programming used for Invoice Hub are:

1. Functions
2. Array
3. Class & Objects
4. File Handling
5. Branching
6. Looping
7. String

# 2. System Requirements

## 2.1 Hardware:

1. 2GB RAM or more.
2. Color Monitor (LCD, LED)
3. Intel i3 or more
4. 64 GB storage.

## 2.2 Software:

1. Operating System: Windows (Windows XP, Windows 7, Windows 8, Windows 10), MAC, Linux

# 3. System design

## 3.1 Algorithm

## A. Program Initialization

1. **Include Necessary Libraries:**
   * <iostream>, <fstream>, <conio.h>, <windows.h>, <iomanip>, <limits>, <ctime>.
2. **Class Definitions:**
   * shopping for product management and billing.
   * permission for user authentication.

## B. User Authentication (Class permission)

### a. Checking Account Existence:

* Open users.txt.
* If a username and password exist, return 1 (account exists); otherwise, return 0 (account does not exist).

### b. Sign-Up Process:

* Check if an account already exists:
  + If yes, redirect to the login process.
  + If no, prompt for a new username and password.
* Store credentials in users.txt.
* Confirm successful signup and redirect to loggedInMenu().

### c. Login Process:

* Check if users.txt exists.
* Prompt for a username and password.
* If credentials match, return 1 (successful login); otherwise, allow up to **three attempts**.
* If login fails three times, return 0 (failed login).

## C. Main Menu (Class shopping)

### a. Display Options:

1. Sign Up
2. Log In
3. Exit

### b. Handle User Input:

* If 1 is chosen, call signUp().
* If 2 is chosen, call logIn() and navigate to loggedInMenu().
* If 3 is chosen, exit the program.

## D. Logged-in Menu (Class shopping)

### a. Display Options:

1. Admin (Modify Products)
2. Buyer (Generate Invoice)
3. Exit

### b. Handle User Input:

* If 1 is chosen, call admin().
* If 2 is chosen, call buyer().
* If 3 is chosen, exit the system.

## E. Admin Menu (Modify Products)

### a. Display Options:

1. Add Product
2. Edit Product
3. Delete Product
4. Back to Main Menu

### b. Handle User Input:

* If 1 is chosen, call add().
* If 2 is chosen, call edit().
* If 3 is chosen, call rem().
* If 4 is chosen, return to loggedInMenu().

## F. Buyer Menu (Purchase & Invoice Management)

### a. Display Options:

1. Buy Product (Generate Receipt)
2. Show All Invoices
3. Search Invoice by Customer Name
4. Go Back
5. Exit

### b. Handle User Input:

* If 1 is chosen, call receipt().
* If 2 is chosen, call showAllInvoices().
* If 3 is chosen, call searchInvoice().
* If 4 is chosen, return to loggedInMenu().
* If 5 is chosen, exit the system.

## G. Product Management (Admin Features)

### a. Adding a Product (add()):

* Prompt the admin to enter product details (code, name, price, discount).
* Check if the product code already exists in database.txt.
* If not, append the new product details.
* Display confirmation.

### b. Editing a Product (edit()):

* Prompt for the product code.
* Search database.txt.
* If found, allow updating name, price, and discount.
* Store changes in a temporary file and rename it back to database.txt.

### c. Removing a Product (rem()):

* Prompt for the product code.
* Search database.txt.
* If found, remove it from the list.
* Store changes in a temporary file and rename it back to database.txt.

## H. Display Product List (list())

* Open database.txt.
* If no products exist, display **"No products available."**
* Otherwise, display product details (code, name, price, discount).

## I. Invoice Management (Buyer Features)

### a. Generating a Receipt (receipt()):

* Ask for the customer’s name.
* Display available products using list().
* Allow selection of multiple products and quantities.
* Calculate total amount with discounts.
* Save invoice details in invoice.txt.

### b. Displaying All Invoices (showAllInvoices()):

* Open invoice.txt and display all invoices.
* If none exist, show **"No invoices found!"**

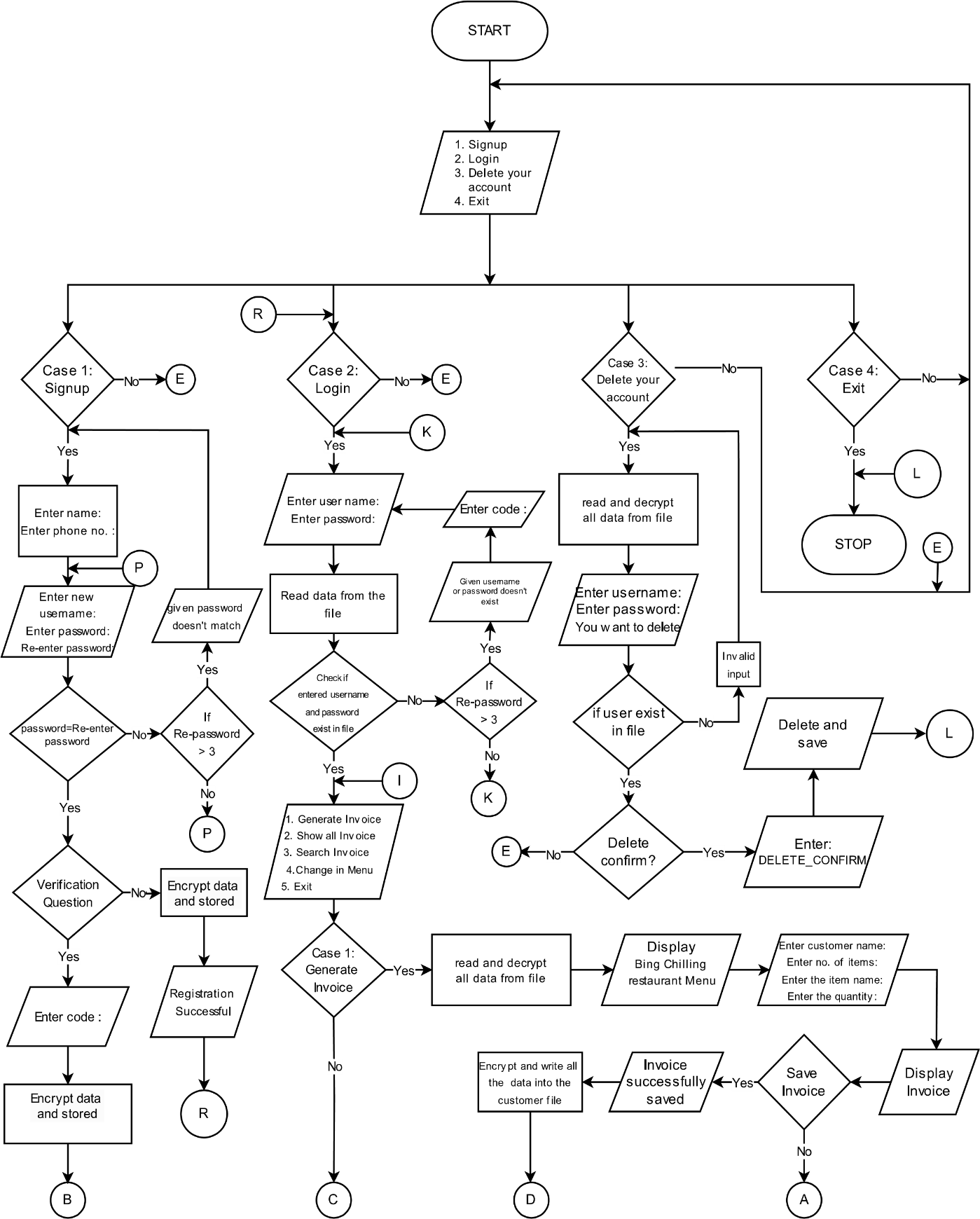
### c. Searching for an Invoice (searchInvoice()):

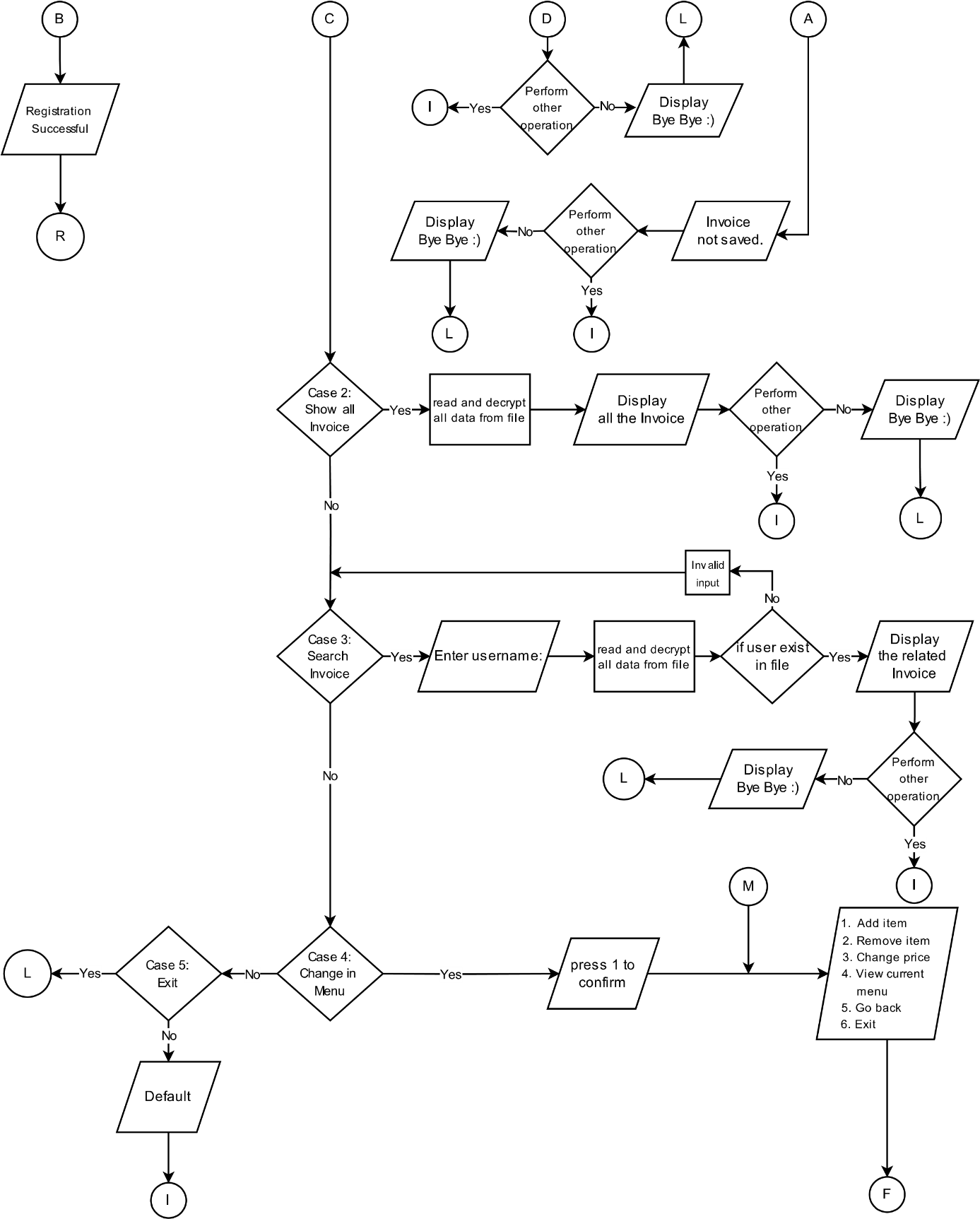
* Prompt for a customer name.
* Validate name format (letters only).
* Search invoice.txt for matching entries.
* Display results or show **"No matching invoice found."**

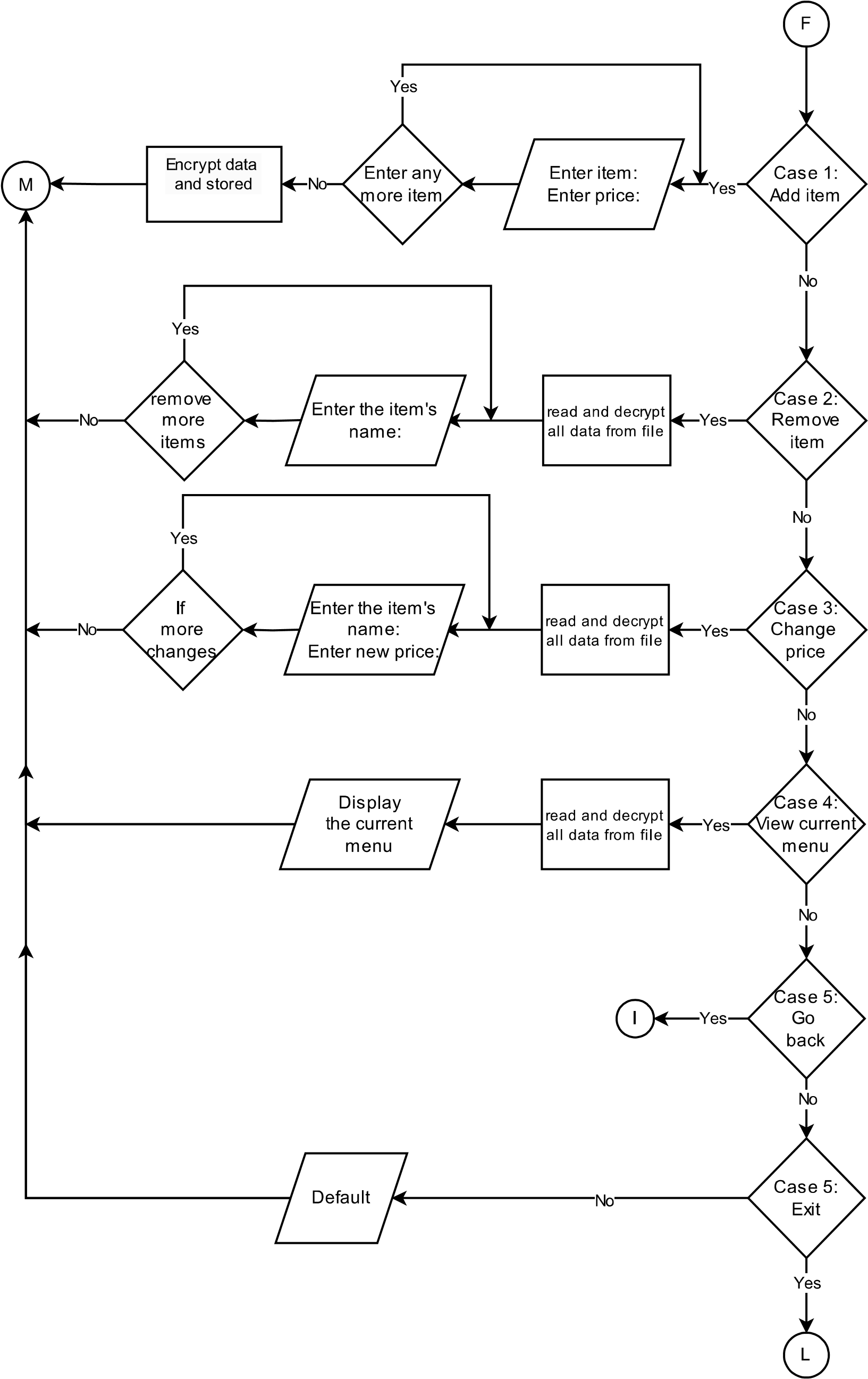
## J. Exit the Program

* If the user selects exit from any menu, display **"Thank you!"** and terminate the program.

## 3.2 Flowchart







# 4. Source Code

https://github.com/Pratik167/Project\_2/tree/main/Second%20Sem%20Project

# 5. System Analysis

 **User Authentication:**

* **Sign-Up:** New users can easily create an account to get started.
* **Login:** Existing users can securely log in to access their accounts.

 **Menu Management:**

* **Add Items:** Staff can quickly add new items to the menu, specifying names and prices.
* **Update Items:** Prices and details of menu items can be updated effortlessly.
* **Remove Items:** Outdated or unavailable items can be removed from the menu with ease.

 **Invoice Generation:**

* **Record Orders:** Customer orders are recorded and invoices are generated automatically.
* **Store Invoices:** Invoices are saved for future reference and easy retrieval.

 **Data Security:**

* **Password Encryption:** User passwords are securely encrypted for safety.
* **Secure Data Handling:** All customer and menu data is handled securely to prevent unauthorized access.

 **User Interface:**

* **Ease of Use:** The system is designed to be intuitive and user-friendly, making it easy for restaurant staff to navigate and use.
* **Flexibility:** Staff can efficiently manage menu items and process invoices without hassle.

 **Error Handling:**

* **Accurate Billing:** Automating invoice generation reduces billing errors.
* **Data Validation:** Ensures that all input data is accurate and reliable.

 **Future Enhancements:**

* **Cloud Integration:** Plans to add cloud features for data backup and remote access.
* **UI Improvements:** Ongoing efforts to refine the user interface for an even better user experience.
* **IoT Integration:** Exploring ways to incorporate IoT technology for further optimization of restaurant operations.

# 6. Problems Faced

**1. Compatibility Issues:**

* Different versions of our development environment caused some features to work perfectly in one version but fail in another, making it hard to get everything running smoothly.

**2. Debugging Challenges:**

* Tracking down and fixing bugs was tougher than expected. Each feature, from generating invoices to managing the menu, had its own set of issues that required significant time and effort to resolve.

**3. User Authentication:**

* Setting up a secure system for user login and password encryption was challenging. Ensuring only authorized users could access the system required careful planning and implementation.

**4. User Interface Design:**

* Creating a user-friendly interface for restaurant staff took several iterations. Balancing functionality with simplicity required multiple redesigns based on feedback.

**5.** **Testing and Validation:**

* We had to test the system thoroughly to ensure all features worked correctly in different scenarios. This involved a lot of trial and error to make sure we hadn’t missed anything.

.

# 7. Conclusion

Our Invoice system program in C++ successfully creates a simple and efficient way to handle billing, reducing mistakes and making the process faster. Key features include easy bill generation, a login system, and a user-friendly interface. We've tested the system thoroughly and added encryption to keep data safe. In the future, we plan to add more features, improve the design, make it scalable, and compatible with variety of businesses. This project has taught us a lot and shown that our billing system can greatly improve how billing is done.

# 8. References

ADV. LEARNING, YouTube, (Jul. 05, 2021) Password encryption

Retrieved from https://www.youtube.com/watch?v=xtk14TcWaN4&amp;t=1s (accessed Jun 13, 2024).

S.FOX( Apr. 27, 2014  ) delete a structure from a binary file

Retrieve from https://stackoverflow.com/questions/23323723/how-do-i-delete-a-structure-from-a-binary-file (accessed Jun. 18, 2024).

R.VERMA(Nov. 30,2017) Nested structure

Retrieve from https://www.quora.com/What-is-a-nested-structure-and-explain-with-examples-with-syntax (accessed Jun. 15, 2024).