*N.B.K.R Institute of Science &Technology*

*Vidyanagar-Tirupati Dt*

*Cafeteria Management*

***Course : Data Structures***

***Branch: Computer Science***

***Section:D***

***Year : I***

***Semester : II***

***Submitted to : Ashok Selva Kumar***

***Submitted By:***

* ***Sayed.Muskan -24KB1A05GS***
* ***Sayed.Esha parwaz -24KB1A05KH***
* ***Yarasi.Bhuvaneswari -24KB1A05PB***

**Abstract :**

**The Cafeteria Management System is a C language-based project designed to manage cafeteria operations efficiently. It handles user and admin functionalities, including login, menu management, order placement, billing, and balance deduction. The system uses file handling to persist menu and user data and linked lists for dynamic data management. Admins can add, delete, or update menu items and register new users. Users can log in using their card numbers to view the menu, place orders, and check their balance. The project offers an interactive command-line interface. It automates billing and inventory management, improving service speed. The goal is to digitize and simplify the cafeteria experience.**

**Introduction:**

**Today, automation in service industries like cafeterias improves both customer experience and operational efficiency. We chose to develop a Cafeteria Management System because of its real-world applicability and the opportunity to integrate core C programming concepts. Our project idea revolves around automating manual processes, reducing human error, and making cafeteria services faster. It handles both administrative tasks (like managing menu and users) and customer tasks (like placing orders and payments). We focused on simple yet powerful tools like linked lists and file management to maintain data dynamically and permanently. This project strengthens our coding,problem-solving, and system design skills.**

**Objective:**

**The main objective of this project is to automate the basic operations of a cafeteria. We aim to allow users to place orders, view their balance, and pay automatically by deducting amounts from a preloaded cafeteria card. The administrator should be able to add, delete, and modify menu items and manage user registrations easily. The system should maintain data consistency even after program termination by saving data in files. We also aimed to ensure the application is user-friendly, fast, and reliable. Another objective was to build the system modularly, enabling easy future expansions. Overall, the goal was to create a complete, small-scale digital cafeteria**.

**System Requirements:**

**Software Requirements**:

* **Code::Blocks IDE or Turbo C++ Compile**
* **Windows/Linux Operating System (or online C compiler)**

**Hardware Requirements:**

* **Minimum 2 GB RAM**
* **Intel Core i3 Processor or higher**
* **100 MB of disk space for storing files**

**Literature Review:**

**Several systems inspired the idea behind this project. Cafeteria billing systems are often managed using databases like MySQL and languages like Java or Python. However, simpler embedded systems sometimes use lightweight solutions like C. We reviewed several projects where linked lists were used for in-memory data management and file handling ensured data persistence. Basic admin-user systems were common in banking or library management projects. The cafeteria systems often lacked modular file management in C, so we aimed to address that. We referred to online tutorials, textbooks on C programming, and previous cafeteria management system models for building an improved version.**

**Methodology:**

**We followed a step-by-step plan to build this project:**

**1. Finalized project requirements and flow.**

**2. Designed the data structures (MenuItem and User using linked lists).**

**3. Implemented file handling to save and load users and menu items.**

**4. Built the user login and admin login system.**

**5. Developed features like menu viewing, item adding, updating, and deleting.**

**6. Programmed order placement and automatic balance deduction.**

**7. Integrated error handling for invalid inputs and insufficient balances.**

**8. Conducted manual testing by simulating multiple users and orders.**

**9. Optimized code for readability and modularity.**

**10. Prepared documentation and tested the final version extensively.**

**Project Description:**

**Problem Statement:**

**Manual cafeteria operations are time-consuming, error-prone, and inefficient, especially in handling billing and stock updates.**

**Proposed Solution:**

**A C-based digital Cafeteria Management System that automates login, order placement, billing, balance deduction, and menu management.**

**Key Features:**

* **Admin and user login.**
* **Menu viewing and updating.**
* **Real-time order placing.**
* **Automatic inventory updating.**
* **File handling to persist data.**
* **Balance management for users.**

**Flowchart/Algorithm**

**Flowchart:**

**Start**

**|**

**Load Menu and Users**

**|**

**Display Main Menu**

**|**

**+---> User Login ---> User Functions (View Menu, Take Order, View Balance, Logout)**

**|**

**+---> Admin Login ---> Admin Functions (View Menu, Add/Update/Delete Menu, Add User, Logout)**

**|**

**Repeat until Exit**

**|**

**Save all data**

**|**

**End**

**Algorithm:**

**1. Load menu and user data from files.**

**2. Display login options (User/Admin).**

**3. If User:**

**Authenticate by card number.**

**Allow to order, view menu, check balance.**

**4. If Admin:**

**Authenticate by PIN.**

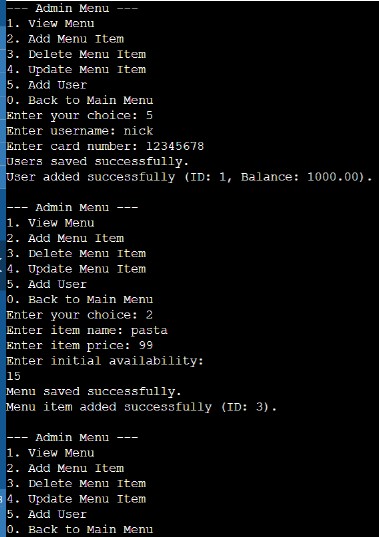
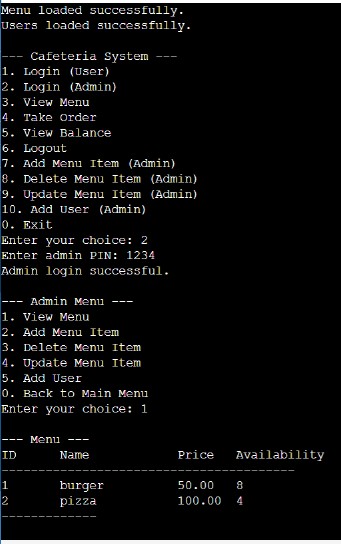
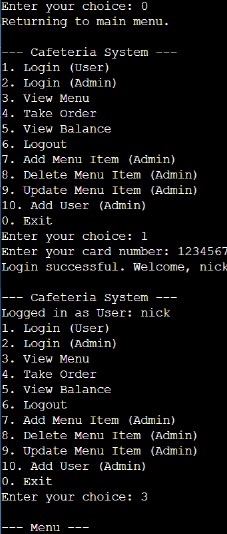
**Allow to add/update/delete menu items or add users.**

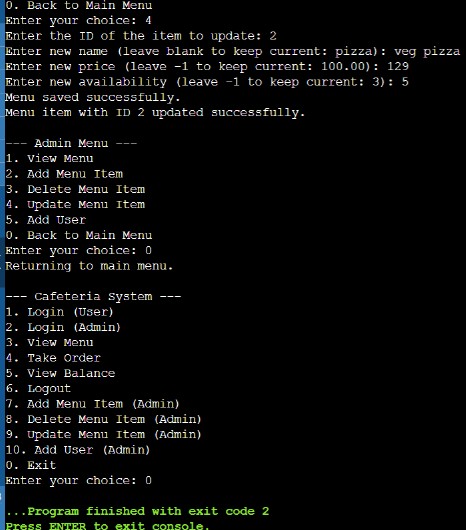
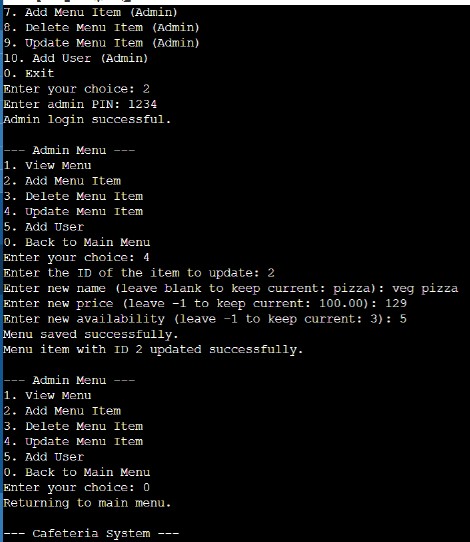
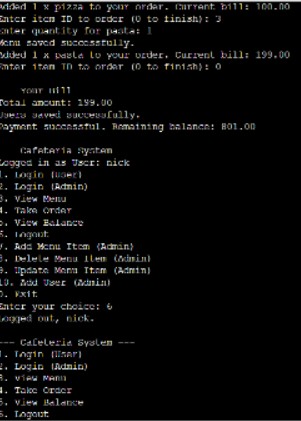
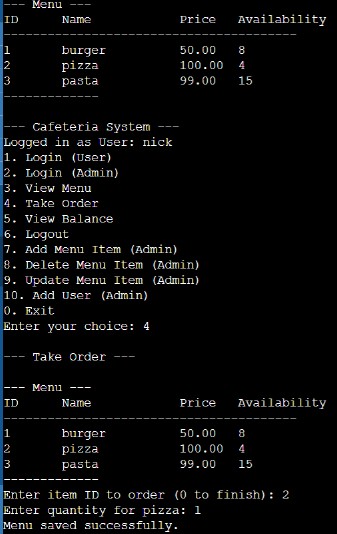
**5. Save changes after every operation.**

**6. Exit program gracefully, freeing all memory.**

[**Source Code Link**](https://onlinegdb.com/67uB0mWfL)

**Output screenshots:**

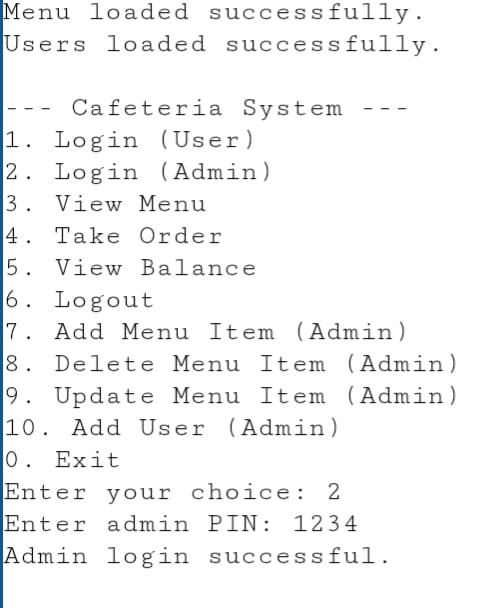
** **

****

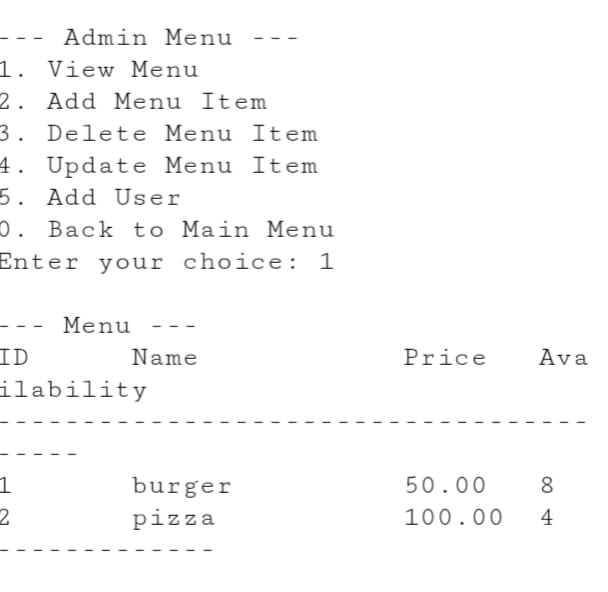
**Testing and validation:**

**Test-1: Admin login**

**Sample output:**

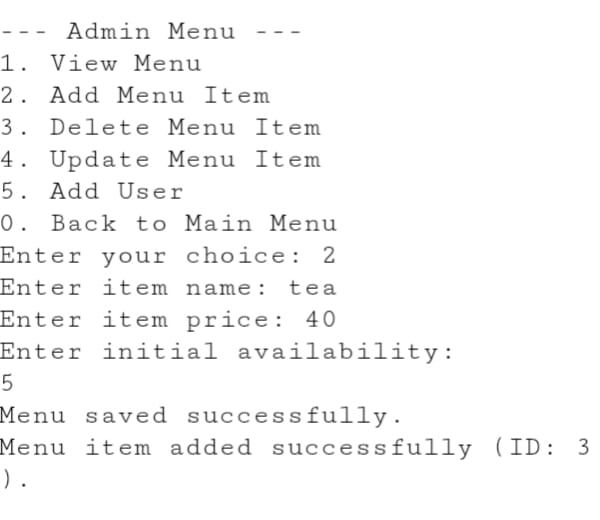
****

**Test 2 :View menu**

**Sample output** :****

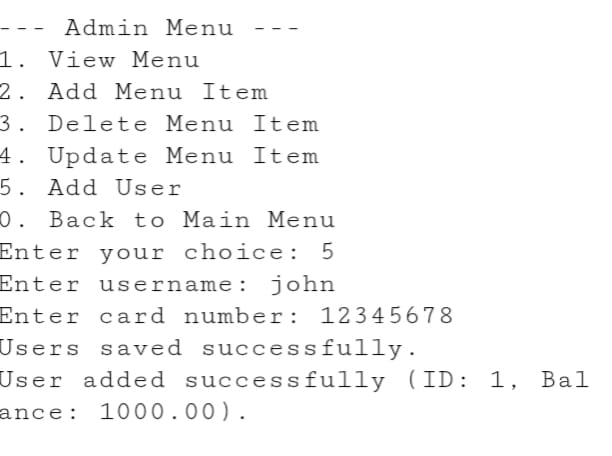
**Test 3: Add menu**

**Sample output:**

****

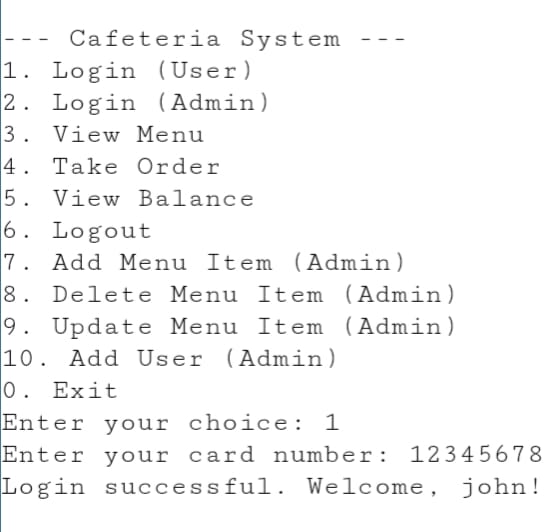
**Test 4:Add user**

**Sample output:**

****

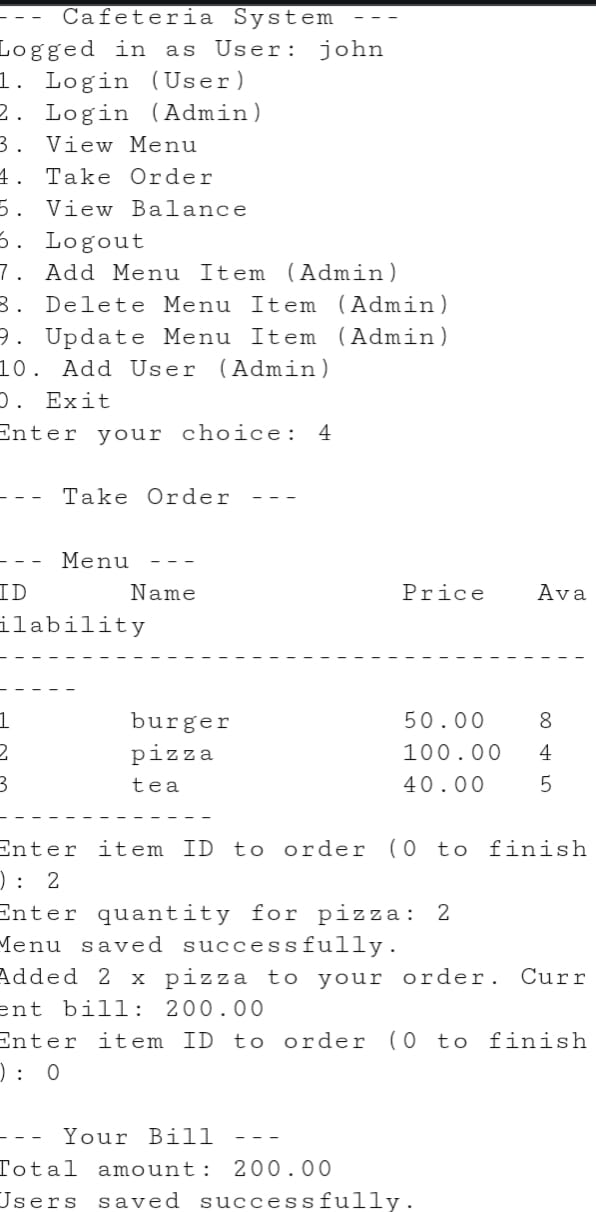
**Test 5:User login**

**Sample output:**

****

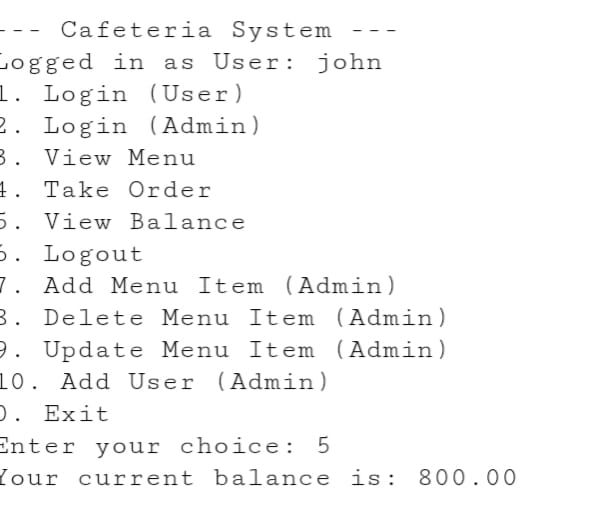
**Test 6:Take order**

**Sample output:**

****

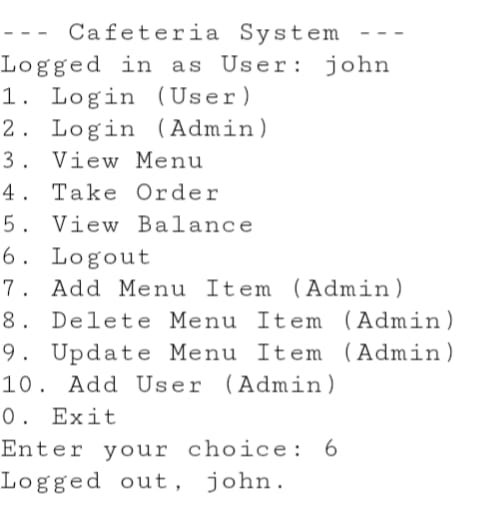
**Test 7:View balance**

**Sample output:**

****

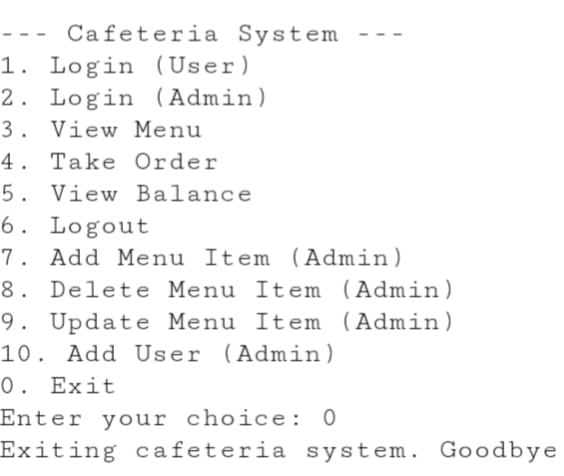
**Test 8:User logout**

**Sample output:**

****

**Test 9:Exit**

**Sample output:**

****

**Limitations:**

* **The project currently works on the command-line interface; no GUI.**
* **File corruption or manual file edits can cause issues.**
* **Admin authentication uses a hard-coded PIN (not very secure).**
* **No support for multiple concurrent users (single-session handling only).**
* **No advanced reports (like sales history, popular items).**
* **Limited to small cafeteria scale (no database management).**
* **No password reset for admin/user.**
* **Not optimized for very large data (menu/user count).**
* **Order history for users is not maintained.**
* **Minor UI improvements like confirmation prompts could be added.**

**Future Enhancements:**

* **Add a graphical user interface (GUI) for better user experience.**
* **Implement a more secure login system (password encryption).**
* **Integrate sales reports and analytics.**
* **Introduce a database backend (e.g., MySQL).**
* **Enable multi-user access and session management.**
* **Add SMS or email notifications for balance low alerts.**
* **Develop mobile app integration for the cafeteria.**
* **Allow balance top-up and recharge functionality.**
* **Add special discount offers and promotions management.**
* **Create a detailed order history log for users and admin review.**

**Conclusion:**

**The Cafeteria Management System project successfully automated basic cafeteria operations like menu management, billing, and balance tracking. Through this project, we gained practical experience in C programming, especially file handling and dynamic memory allocation using linked lists. We understood how real-world systems maintain data consistency and user sessions. We also learned how to manage program modularity, error handling, and program testing. Despite some limitations, the system meets its core objectives and serves as a solid foundation for future improvements. Overall, the project enhanced our coding, debugging, and software development skills significantly.**

**References:**

1. **Let Us C by Yashavant Kanetkar**
2. **Programming in ANSI C by E. Balagurusamy**
3. **TutorialsPoint C Programming Tutorials.**
4. **StackOverflow (for specific error solutions).**
5. **GeeksforGeeks C File Handling Tutorials.**
6. **Online sample projects on cafeteria systems.**

**Acknowledgement:**

**We would like to express our sincere gratitude to our project guide and faculty members for their continuous support and encouragement throughout the development of this Cafeteria Management System. Their valuable guidance helped us understand the importance of effective system design and logical implementation. We also thank our peers for their suggestions and motivation. This project enhanced our understanding of file handling, linked lists, and user interface logic in C programming. We acknowledge the efforts of open resources and online communities that provided technical insights when required. We learned the importance of testing, debugging, and documenting a software project. Lastly, we appreciate the opportunity to apply theoretical knowledge practically.**

THANK YOU