NARSIMHA REDDY
ENGINEERING COLLEGE
An Autonomous Institution| Affiliated to JNTUH | Approved by AICTE
Accredited by NBA & NAAC with 'A' Grade

CodeCraft: Learning C

Through 20 Real-World Programming Projects

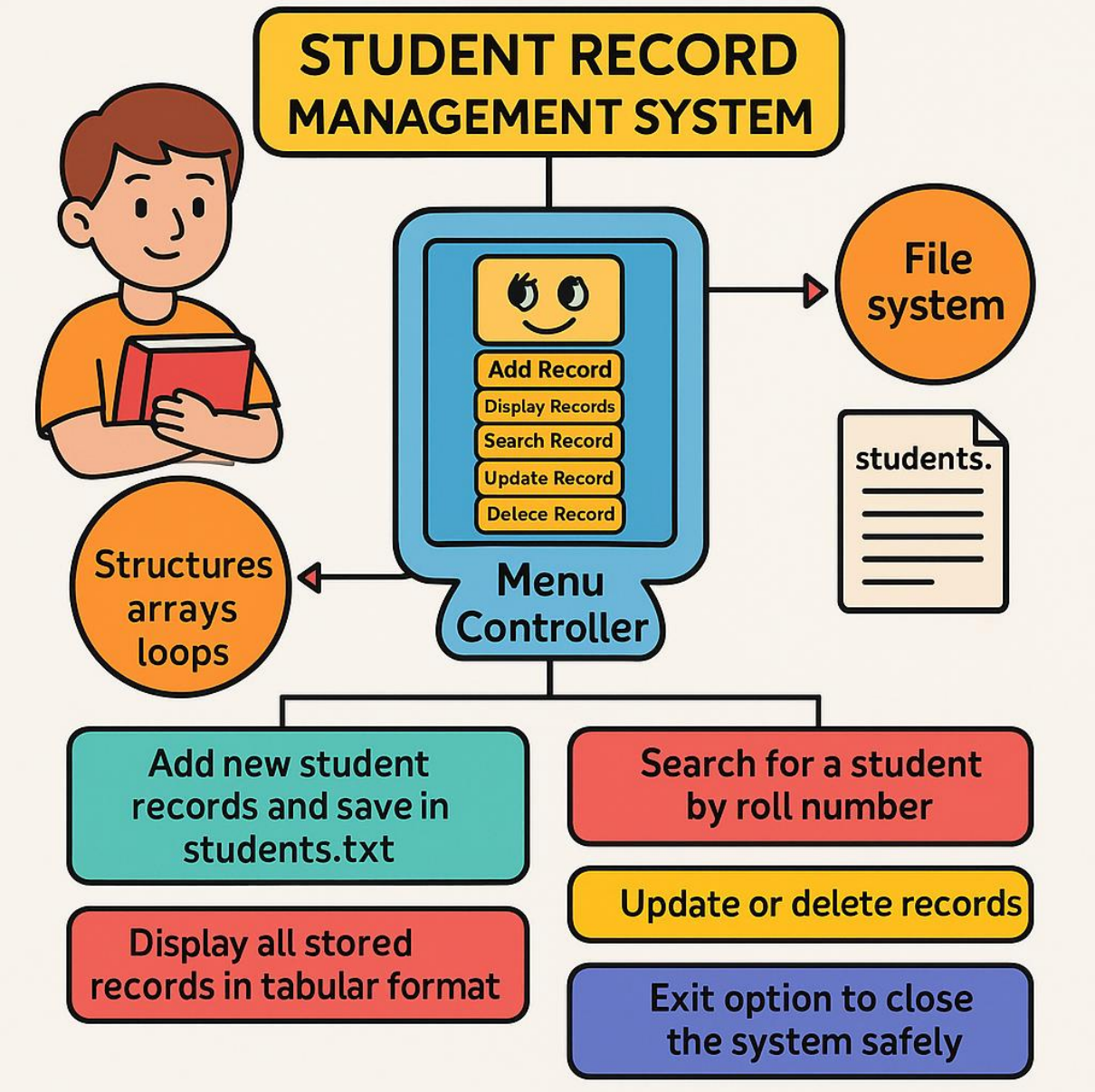
PREPARED BY

CAREER DEVELOPMENT CENTER

C Programming Projects

1. Student Record Management System

In many small educational institutions, student records are still handled manually through registers and Excel sheets. When exam season arrives, teachers scramble to calculate totals and grades, often leading to delays and mistakes. The *Student Record Management System* solves this issue by using C programming to store, retrieve, and modify student details efficiently. The system uses structures to hold individual student data such as roll number, name, marks, and grade. With arrays and functions, it allows adding new students, searching by roll number, and updating marks instantly. File handling ensures every modification is permanently stored in a .txt file, eliminating data loss even if the program closes. This project mirrors a small-scale digital academic ERP, showing how coding can streamline repetitive manual work and bring accuracy to educational data management.



Why This Project?

- Demonstrates structured data organization and file persistence.
- Helps understand CRUD operations using text files.
- Bridges academic problems and programming applications.

Key Features

- Add, search, update, and delete student data.
- Generate grade summaries.
- Store information permanently in *students.txt*.

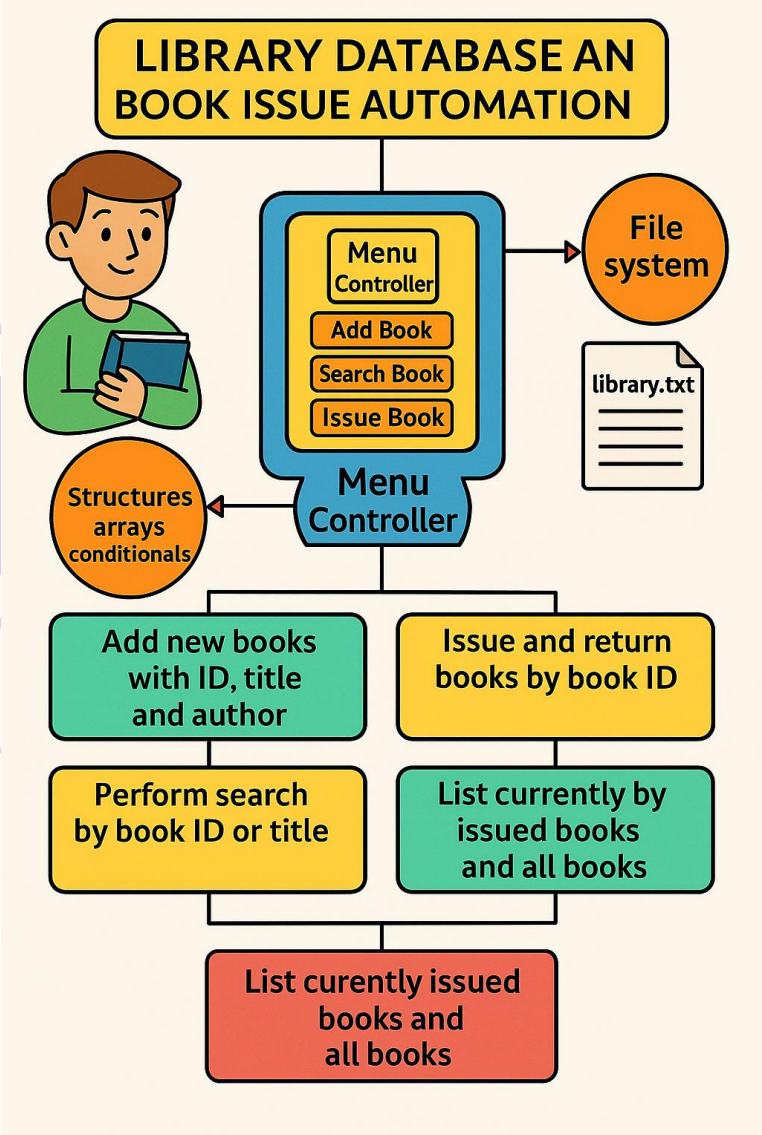
Expected Outcome

Students learn how to use C programming for digital data storage and develop a real-world understanding of record automation.

2. Library Database and Book Issue Automation Using C Programming

Project Description

Libraries in small towns or schools often struggle to track book issues and returns manually. Librarians record details in notebooks, which become messy and inconsistent over time. The *Library Database and Book Issue Automation System* transforms this process into a clean digital workflow. Each book is represented by a structure containing book ID, title, author, and availability status. The librarian can search, issue, and return books through an interactive menu. When a member borrows a book, the system updates its availability and records the issue date in *library.txt*. The use of functions and loops makes the program modular, and arrays allow multiple records to be handled easily. This project not only helps librarians but also demonstrates to beginners how simple code can replicate a real information system that saves time and prevents errors.



Why This Project?

- Illustrates practical database concepts using text files.
- Encourages students to build real-world systems.
- Strengthens modular programming and string handling.

Key Features

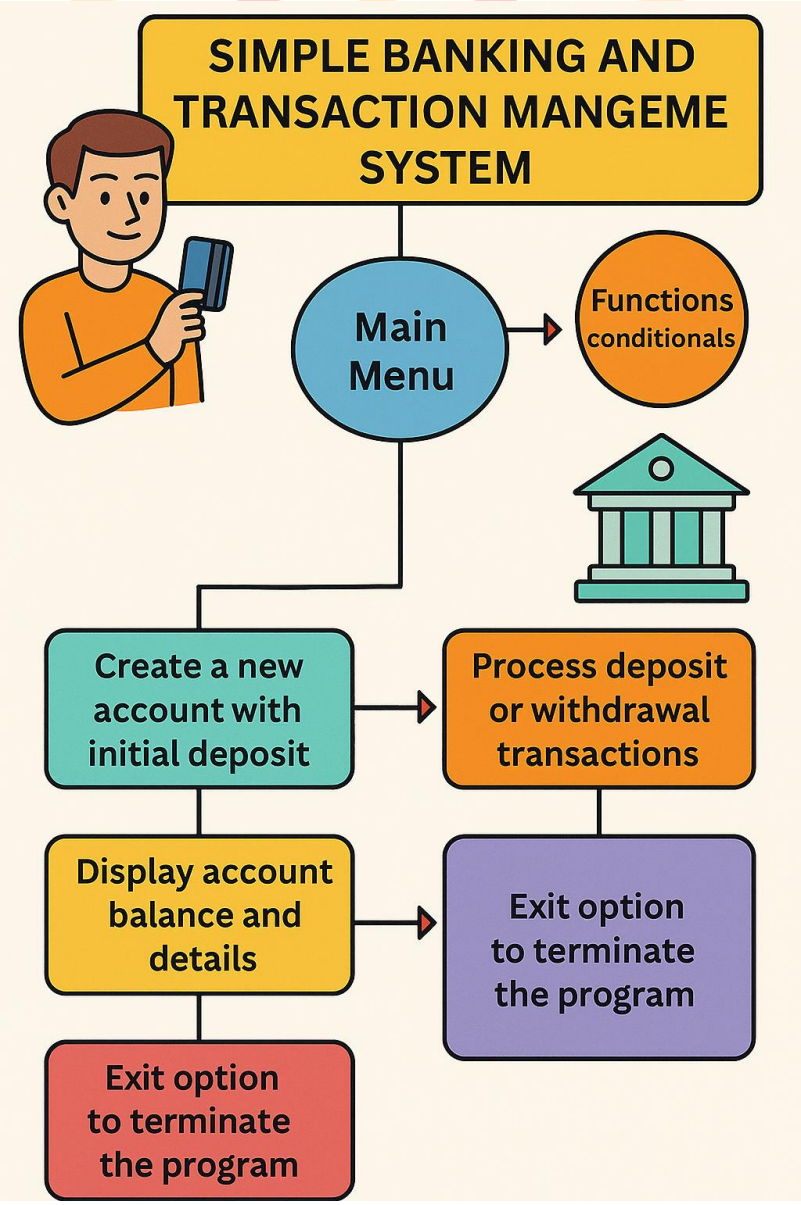
1. Book addition, search, issue, and return modules.
2. Automatic update of availability status.
3. Storage in *library.txt* for long-term use.

Expected Outcome

Learners grasp how to simulate real administrative databases using C and file handling.

3. Simple Banking and Transaction Management System

A local cooperative bank needed an easy way to manage small customer accounts without using expensive software. The *Banking and Transaction Management System* written in C provides a perfect lightweight solution. It allows customers to open an account, deposit money, withdraw funds, and check balances. Every transaction updates the stored record in *bank.txt*. The program relies on loops for transaction cycles, conditionals for validation, and structures for organized storage of account information. Beyond technical skills, this project teaches students the logic behind secure transactions and how each operation in banking must be validated before updating. It's a real-world example of applying C to simulate business workflows that involve numeric data processing and integrity checks.



Why This Project?

- Simulates real-life financial data management.
- Reinforces input validation and data persistence.
- Builds understanding of arithmetic operations in C.

Key Features

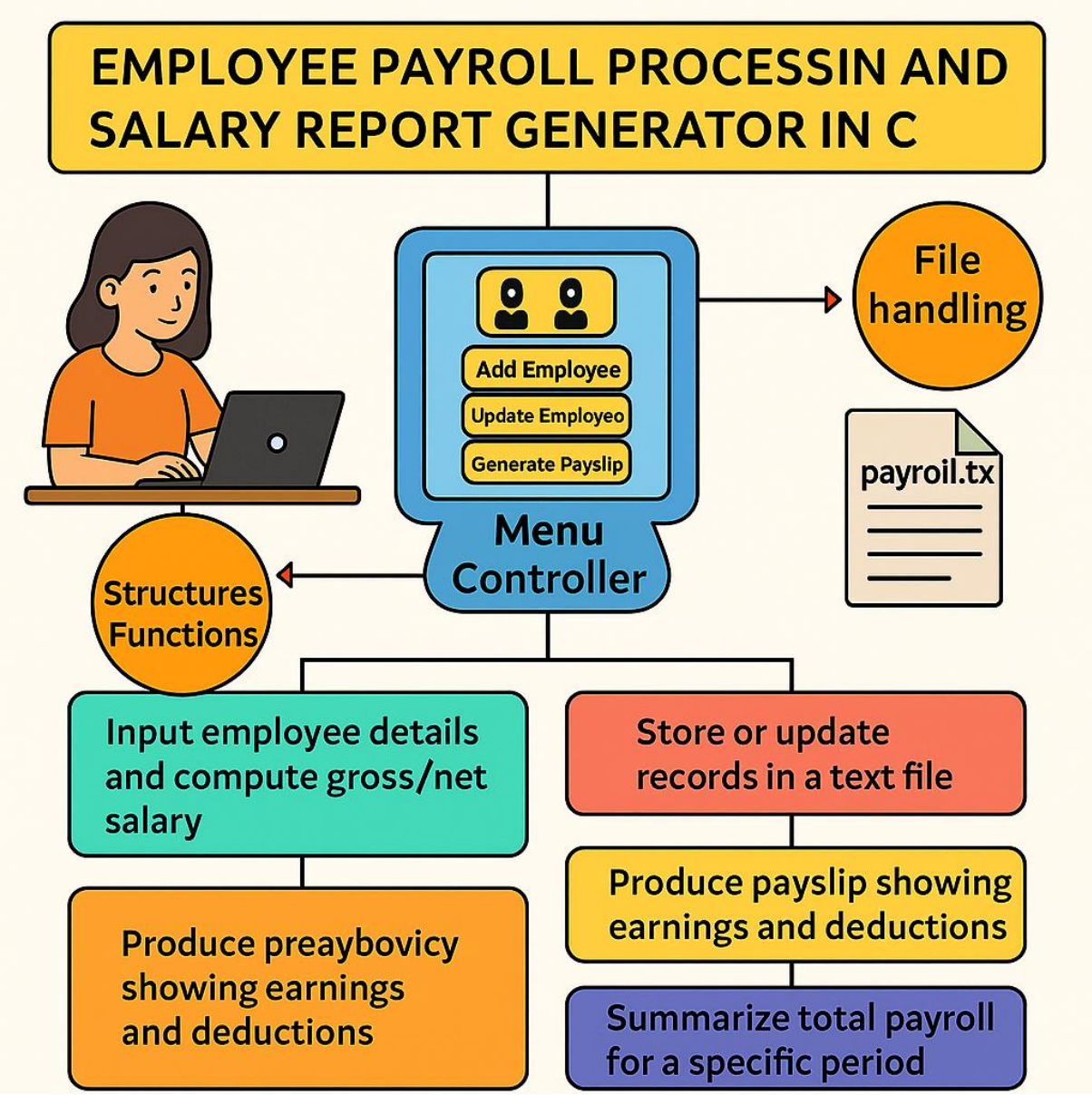
1. Account creation, deposit, and withdrawal.
2. Balance inquiry and modification.
3. Permanent records in *bank.txt*.

Expected Outcome

Students learn structured transaction handling and numeric data management in a business context.

4. Employee Payroll Processing and Salary Report Generator in C

For many small companies, salary computation is still done on paper or in spreadsheets that require manual entry every month. Errors in overtime, deductions, and tax calculations can cause frustration for both staff and management. The *Employee Payroll Processing System* developed in C automates salary management by calculating gross and net pay for every employee. It captures essential data—employee ID, name, designation, basic pay, allowances, and deductions—using structures. Arithmetic functions compute totals and save them in *payroll.txt*. The program is modular, allowing HR staff to easily update details or generate reports by month. Through this project, learners understand how logic and computation combine to create professional solutions for administrative needs.



Why This Project?

- Simulates real HR management functions.
- Demonstrates data organization using structures.
- Reinforces arithmetic and modular programming.

Key Features

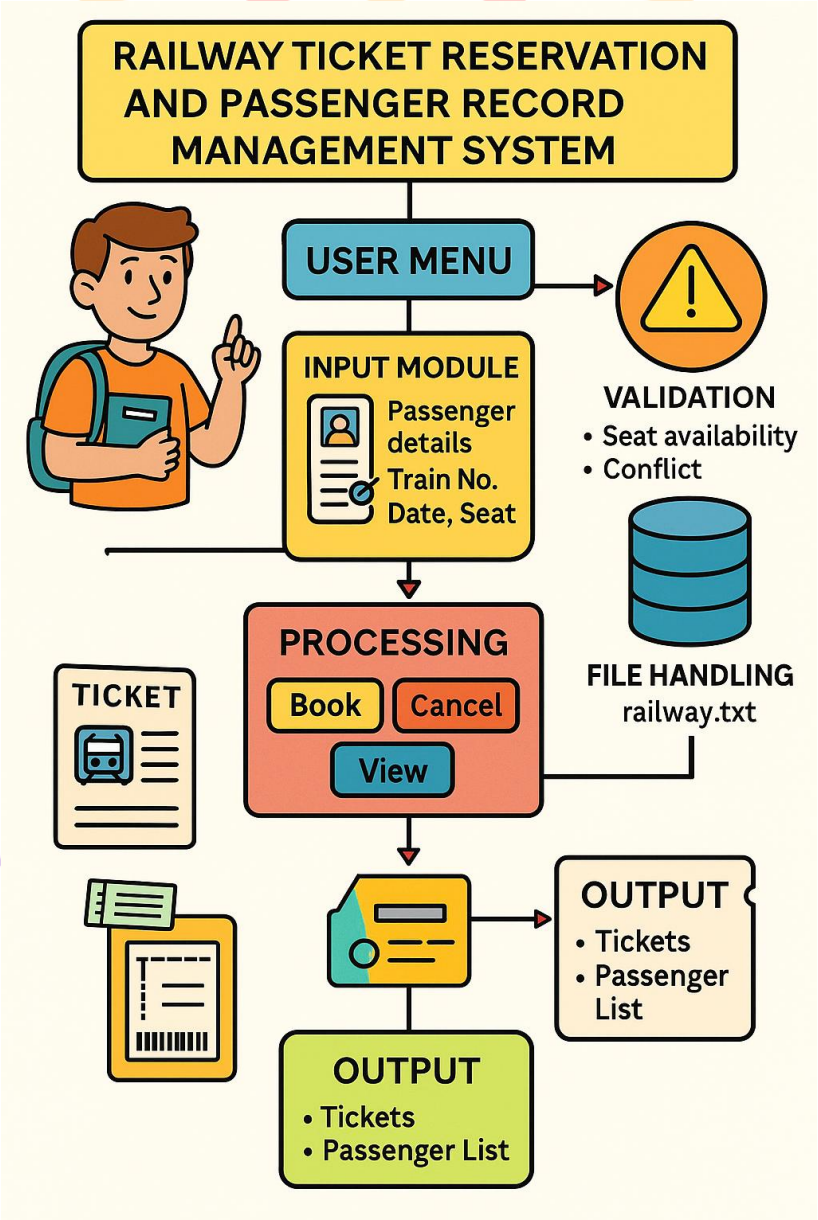
1. Employee data input and modification.
2. Automatic salary computation.
3. Monthly report generation.

Expected Outcome

Learners understand how programming principles support real-life payroll systems efficiently.

5. Railway Ticket Reservation and Passenger Record Management System

Railway booking clerks often face long queues and manual register entries, increasing the risk of double bookings. The *Railway Ticket Reservation System* automates this using C programming. It allows passengers to reserve, cancel, or modify tickets with details like train number, passenger name, and date of travel. All records are stored in *railway.txt* and updated dynamically. Using arrays and file handling, the system ensures that seat availability and booking details remain accurate. The project also provides reports showing total passengers or available seats for each train. It brings students close to the logic used in large railway systems, demonstrating how small-scale simulations can model major real-world operations.



Why This Project?

- Demonstrates reservation logic and validation.
- Introduces multi-record handling through arrays.
- Emphasizes persistent data management.

Key Features

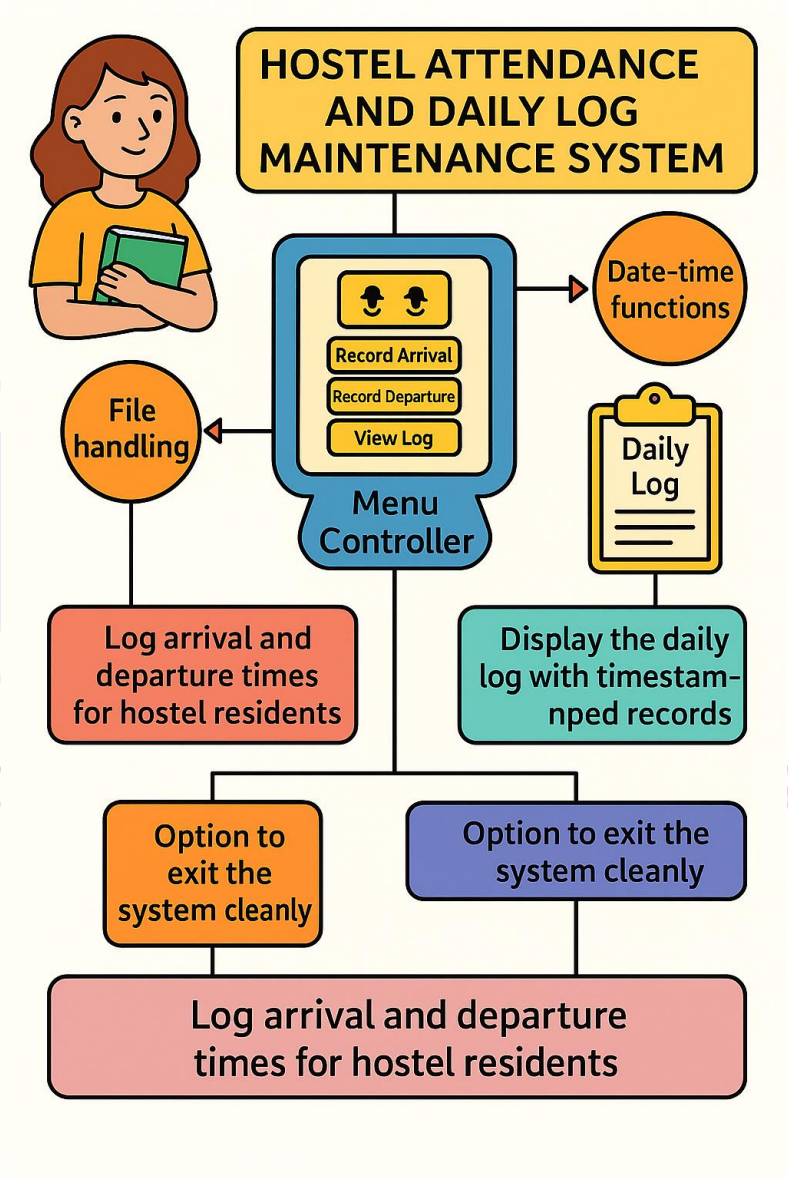
1. Book, cancel, or view tickets.
2. Automatic seat-availability updates.
3. Record saving in *railway.txt*.

Expected Outcome

Students learn structured transaction design and the foundations of transport system automation.

6. Hostel Attendance and Daily Log Maintenance System

In most college hostels, wardens maintain large registers to record attendance. During inspections, finding who was absent on a given date takes hours of searching. The *Hostel Attendance and Daily Log Maintenance System* automates this using the C language. It allows the warden to enter the student's roll number, name, and daily attendance status ("Present" or "Absent") and stores it in *attendance.txt*. The program displays summaries for specific dates or months using loops and conditional logic. Through structures and arrays, multiple entries are managed efficiently, and file handling ensures attendance records are permanently saved for future verification. This project mirrors real hostel workflows, demonstrating how C programming can be used to eliminate repetitive manual recording and improve accuracy in monitoring student discipline and hostel occupancy.



Why This Project?

- Teaches file appending, searching, and updating.
- Encourages system-based attendance management.
- Improves data accuracy and reliability.

Key Features

1. Mark and retrieve daily attendance.
2. Search absentees by date.
3. Save all records in *attendance.txt*.

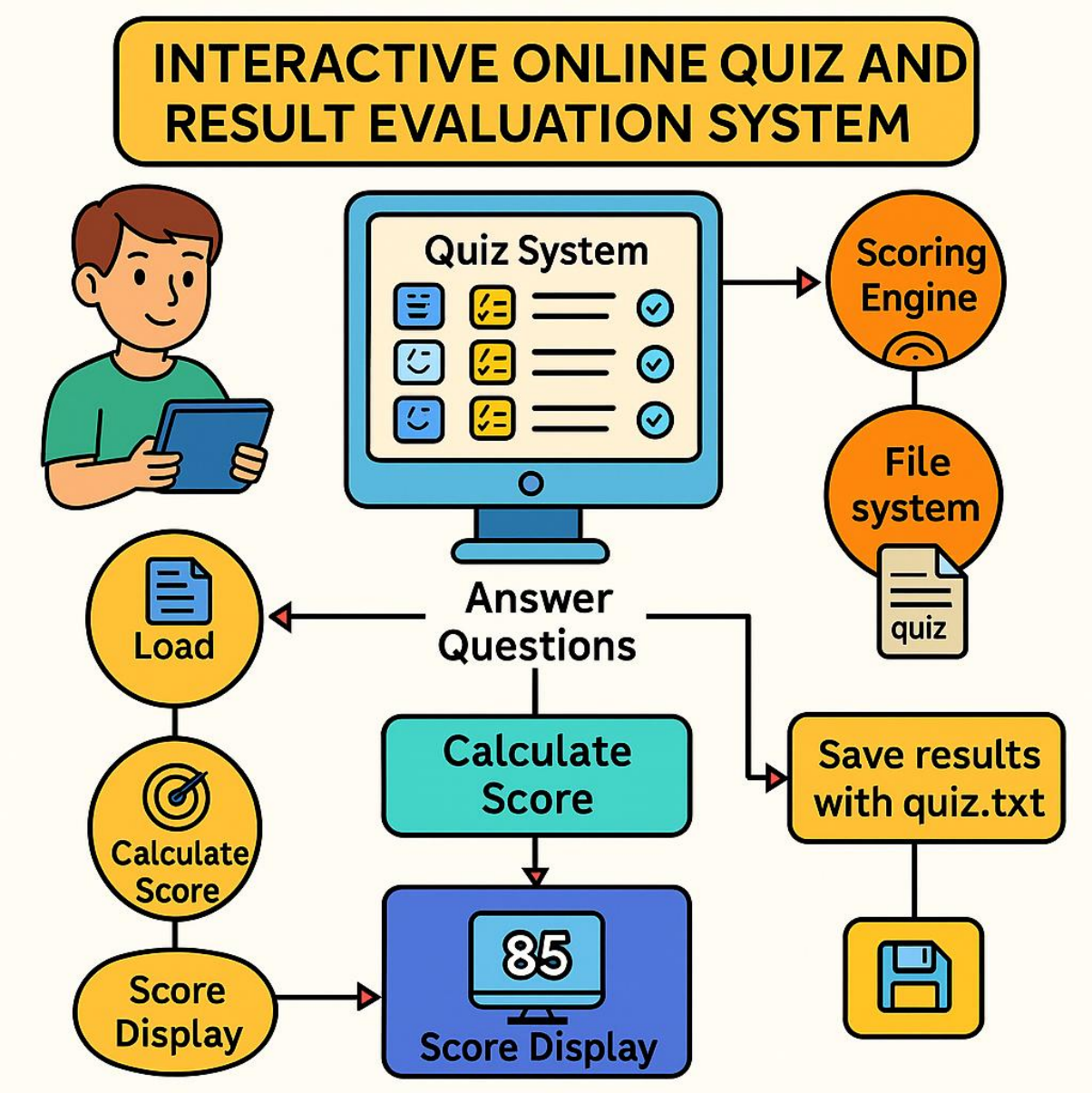
Expected Outcome

Students understand file-based data maintenance and learn to implement iterative logic for real-life tracking systems.

7. Interactive Online Quiz and Result Evaluation System

Project Description

Educational institutions often organize quizzes but rely on manual paper checking, which delays results. The *Online Quiz and Result Evaluation System* built in C brings automation to this process. It reads multiple-choice questions from *quiz.txt*, displays them one by one, accepts user responses, and calculates scores instantly. Each participant's name and score are appended to a result file for future reference. Arrays handle questions and options efficiently, while loops and conditionals control user flow. This project is an engaging way for beginners to apply logic, control structures, and file handling in one integrated program. It also replicates online assessment tools used in e-learning platforms, showing how simple code can simulate large educational systems.



Why This Project?

- Combines logic, arrays, and conditionals.
- Demonstrates interactive programming.
- Builds understanding of file read/write operations.

Key Features

1. Read questions from *quiz.txt*.
2. Display results instantly.
3. Store participant scores permanently.

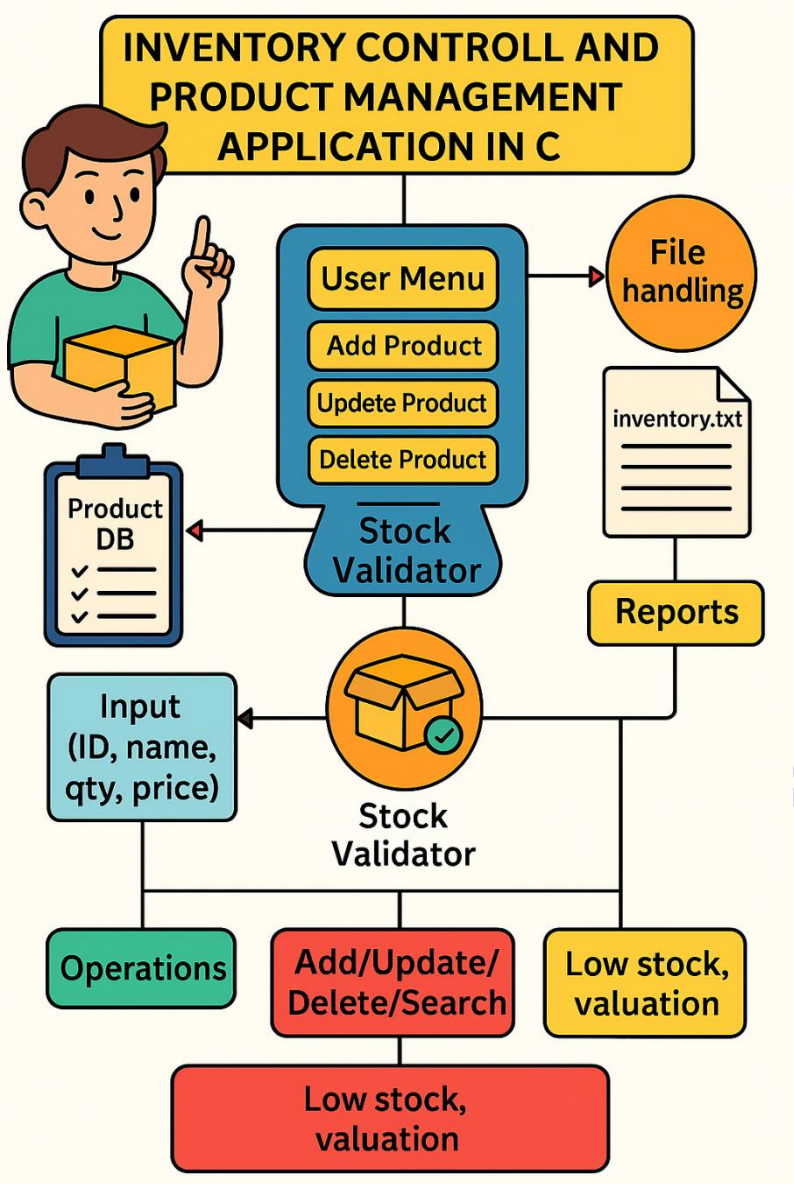
Expected Outcome

Learners acquire hands-on experience with evaluation systems and input-output automation.

8. Inventory Control and Product Management Application in C

Project Description

A small grocery store owner noticed frequent stock mismatches and decided to digitize inventory tracking. The *Inventory Management Application* created in C stores product name, ID, quantity, and price using arrays and structures. The owner can add, update, or delete items through a menu-driven interface. All data is written to *inventory.txt* to maintain stock continuity between sessions. The program automatically calculates total stock value, highlights low-stock items, and allows quick price updates. This project shows how a simple console-based program can make a big difference for small businesses struggling with manual stock entries. It's a practical example of how programming supports entrepreneurship through accuracy and automation.



Why This Project?

- Demonstrates file-based inventory automation.
- Reinforces array and structure concepts.
- Encourages real-world application of arithmetic logic.

Key Features

1. Add/update/delete product data.
2. Display stock report.
3. Store all data in *inventory.txt*.

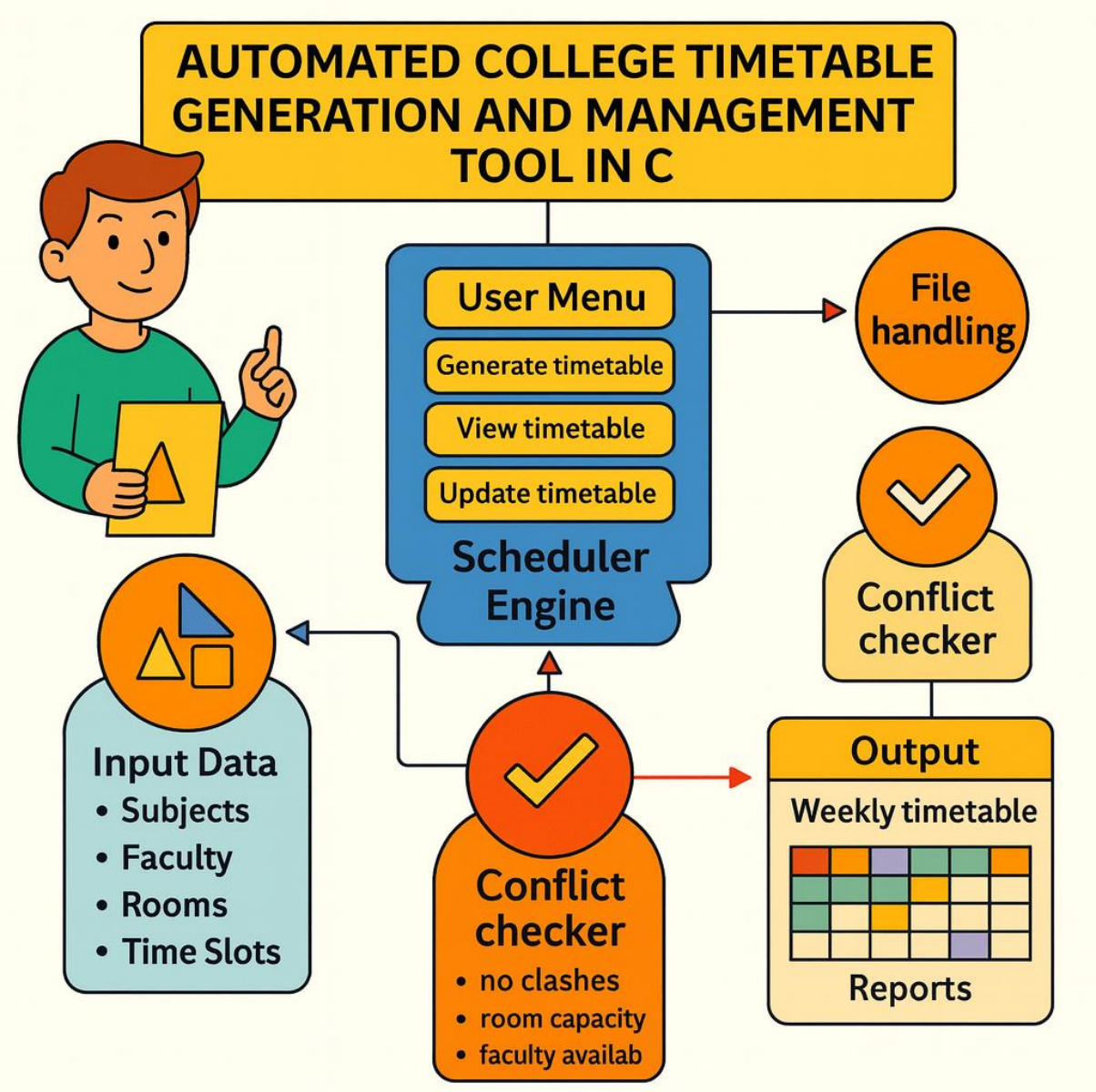
Expected Outcome

Students learn how programming logic can replicate small-scale business software.

9. Automated College Timetable Generation and Management Tool

Project **Description**

College administrators face a major challenge in creating balanced timetables without overlapping subjects or faculty schedules. The *Automated Timetable Management Tool* in C eliminates this headache. It allows administrators to input faculty names, subjects, and available time slots. The system arranges them logically across days using arrays and conditionals to prevent conflicts. The generated timetable is displayed in a formatted structure and stored in *timetable.txt*. This project offers a small-scale glimpse of how scheduling software operates, focusing on data structuring and formatted output. By simulating a real academic workflow, students learn how planning and logic can work together to automate one of the most time-consuming administrative tasks in education.



Why This Project?

- Solves a real academic scheduling problem.
- Demonstrates formatted output and array management.
- Strengthens logical thinking in problem solving.

Key Features

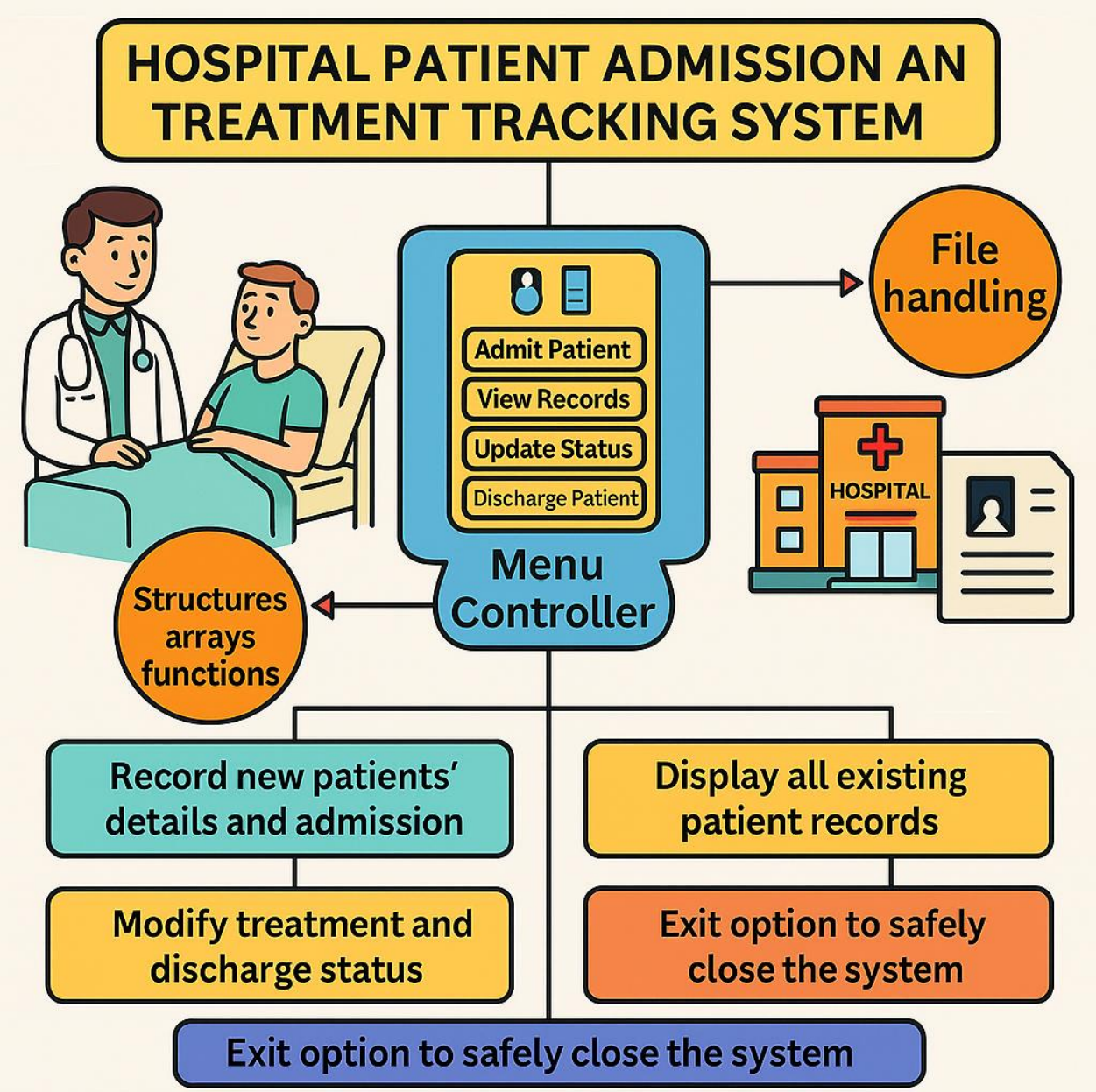
1. Input faculty, subject, and time slots.
2. Generate conflict-free schedules.
3. Store data in *timetable.txt*.

Expected Outcome

Students understand how algorithms help manage time-based academic planning.

10. Hospital Patient Admission and Treatment Tracking System

In small clinics, patient information is often handwritten in registers, causing confusion during follow-ups. The *Hospital Patient Admission and Treatment Tracking System* developed in C brings structure to hospital operations. Each patient's record includes ID, name, age, diagnosis, doctor, and admission date. Data is saved in *hospital.txt* for secure and long-term use. The system allows staff to admit new patients, update treatments, or discharge them when recovered. Using structures, arrays, and file handling, the system simulates a basic version of digital hospital management software. It not only improves data consistency but also helps students visualize how code can impact real-life healthcare operations.



Why This Project?

- Demonstrates structured data handling for healthcare.
- Teaches update and search functions in files.
- Models real-world patient record management.

Key Features

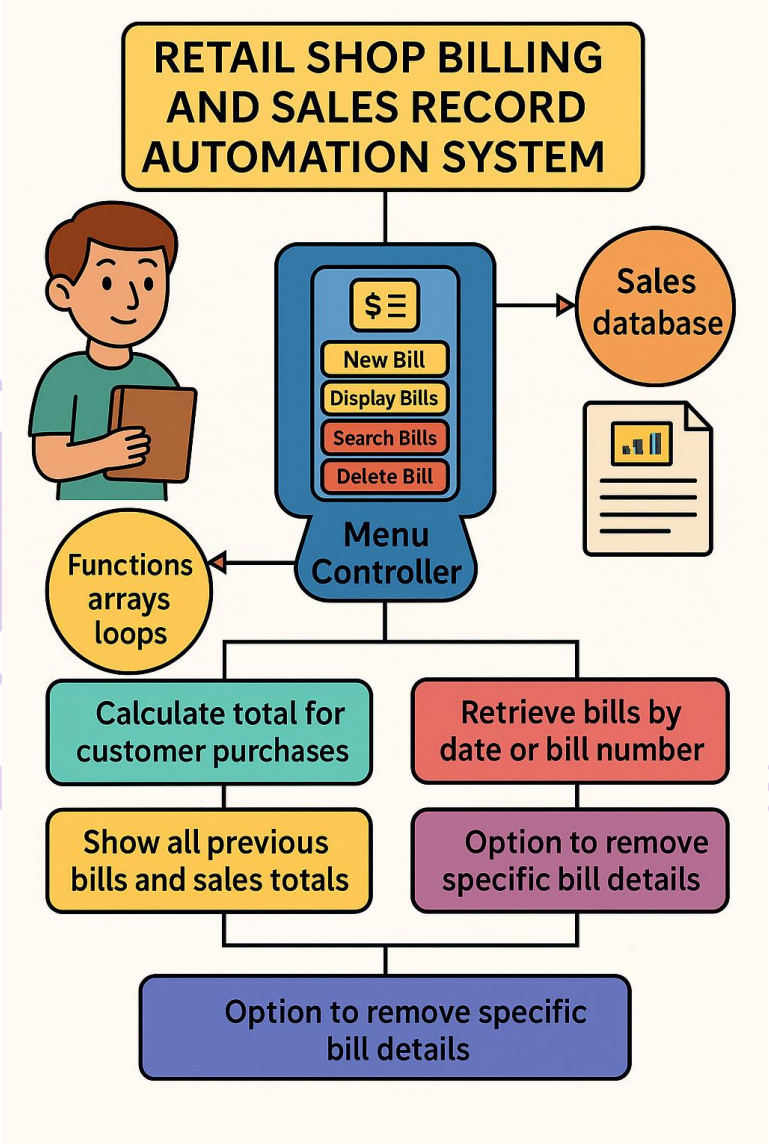
1. Admit, update, and discharge patients.
2. Search records by ID.
3. Save and retrieve from *hospital.txt*.

Expected Outcome

Students learn how programming supports healthcare record organization and operational efficiency.

11. Retail Shop Billing and Sales Record Automation System

In many local retail stores, billing is still done manually using calculators and notebooks, which increases the chances of human error. Customers often have to wait while the shopkeeper recalculates totals or forgets to apply discounts. The *Retail Shop Billing and Sales Record Automation System* simplifies this entire process using C programming. It allows the shopkeeper to input item name, quantity, and price per unit, after which the system automatically calculates the subtotal, discount, tax, and total amount. All the transaction details are saved into *bills.txt* for later reference. The system also supports daily sales reports that help the owner analyze business performance. This project demonstrates how programming can bridge the gap between manual bookkeeping and efficient digital operations, making it a realistic simulation of a small point-of-sale (POS) system.



Why This Project?

- Automates retail billing and record keeping.
- Demonstrates arithmetic and logical operations.
- Improves understanding of file-based reporting.

Key Features

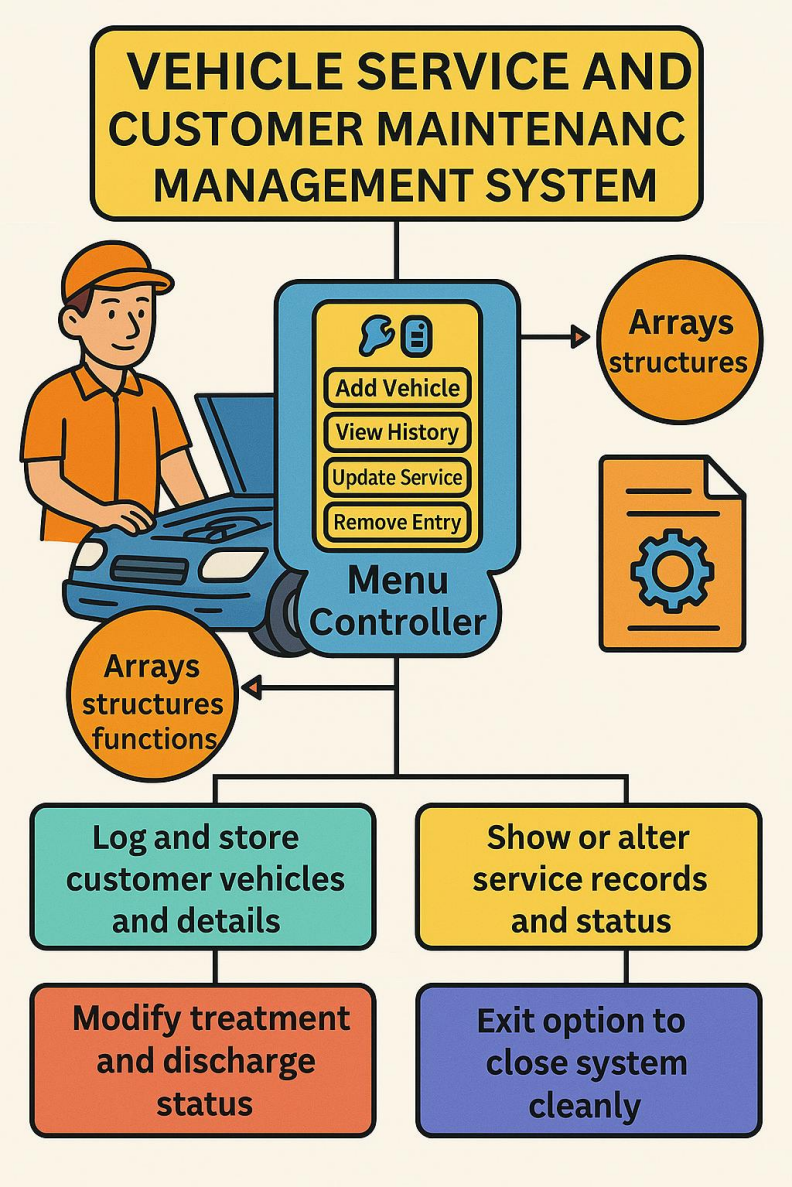
1. Generate itemized bills with totals and discounts.
2. Store all daily transactions in *bills.txt*.
3. Retrieve and view sales summaries.

Expected Outcome

Students understand billing logic, real-world retail automation, and how file handling helps manage commercial data.

12. Vehicle Service Tracking and Customer Maintenance Management System

Automobile workshops handle dozens of vehicles daily, and keeping track of each customer's service record manually becomes cumbersome. The *Vehicle Service Tracking and Customer Maintenance System* built in C is designed to address this problem. It allows the workshop manager to record the vehicle number, owner's name, service type, date, and cost. Using arrays and structures, the system can store multiple records efficiently and update them as services are completed. The data is saved in *service.txt*, allowing quick retrieval when a customer returns for the next maintenance cycle. By simulating the workflow of a real automotive service center, the project teaches file-based record management and how a simple console application can replace handwritten logs.



Why This Project?

- Solves real-world service record tracking issues.
- Demonstrates multi-record storage and retrieval.
- Enhances knowledge of arrays and structures.

Key Features

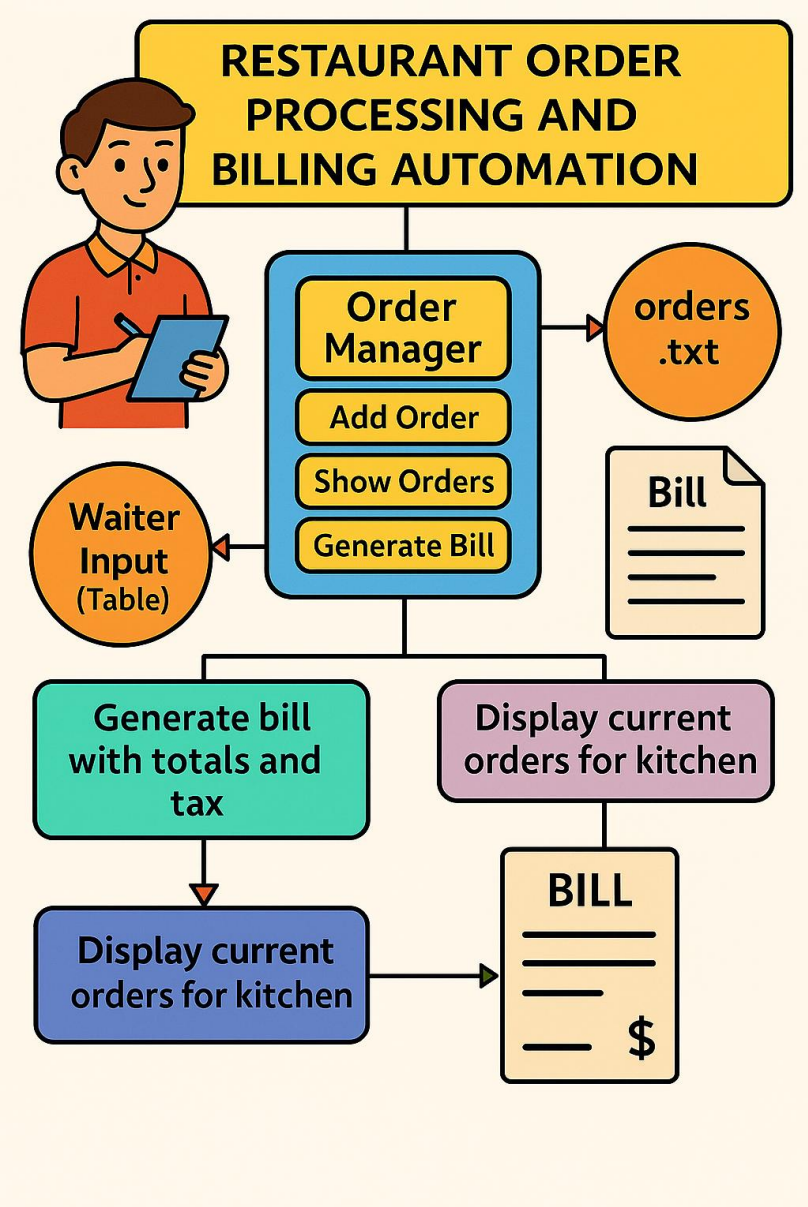
1. Add, update, and view service records.
2. Store all transactions in *service.txt*.
3. Search customers by vehicle number.

Expected Outcome

Students learn how to design efficient service-based management tools using C programming.

13. Restaurant Order Processing and Billing Automation in C

In busy restaurants, waiters often make mistakes while recording orders or calculating totals manually. The *Restaurant Order Processing and Billing Automation System* built in C streamlines the entire dining process. Each table's order can be entered along with item names, quantities, and prices. The system computes totals automatically, generates a formatted bill, and saves all orders in *orders.txt*. The program uses structures and nested loops to handle multiple tables and simultaneous orders. By implementing this project, learners simulate how digital billing systems work in restaurants, replacing manual billing slips with automated accuracy. It's a strong example of applying logic, file handling, and modular programming in a practical real-world domain.



Why This Project?

- Automates the restaurant billing workflow.
- Reinforces arrays, loops, and calculations.
- Encourages modular programming practices.

Key Features

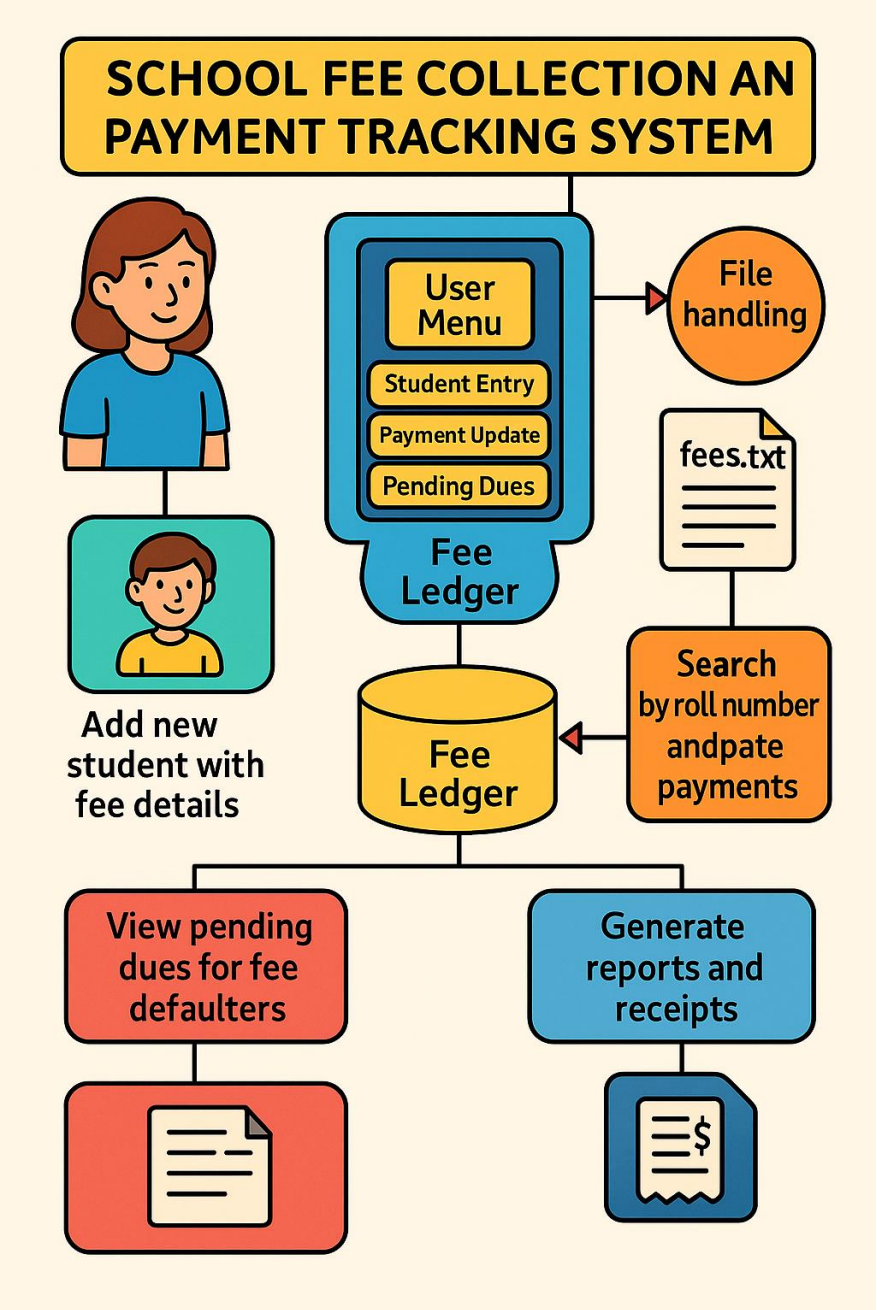
1. Manage orders by table.
2. Compute itemized totals with tax.
3. Save and view daily order history.

Expected Outcome

Students gain experience building menu-driven, interactive applications for commercial use.

14. School Fee Collection and Payment Tracking System

Managing fee payments for hundreds of students is a major challenge in schools, especially when done on paper. The *School Fee Collection and Payment Tracking System* in C automates this process efficiently. It captures each student's name, roll number, class, and payment details such as total fee, amount paid, and balance due. All information is stored in *fees.txt*, ensuring accuracy and continuity. Staff can easily check who has pending fees or generate monthly fee collection reports. Loops and conditionals handle validation, while file handling ensures permanent data storage. This project models the fee management module found in school ERPs and is an ideal example of how C can solve real administrative problems.



Why This Project?

- Solves fee tracking issues in education systems.
- Teaches data updating and search functionalities.
- Reinforces modular programming logic.

Key Features

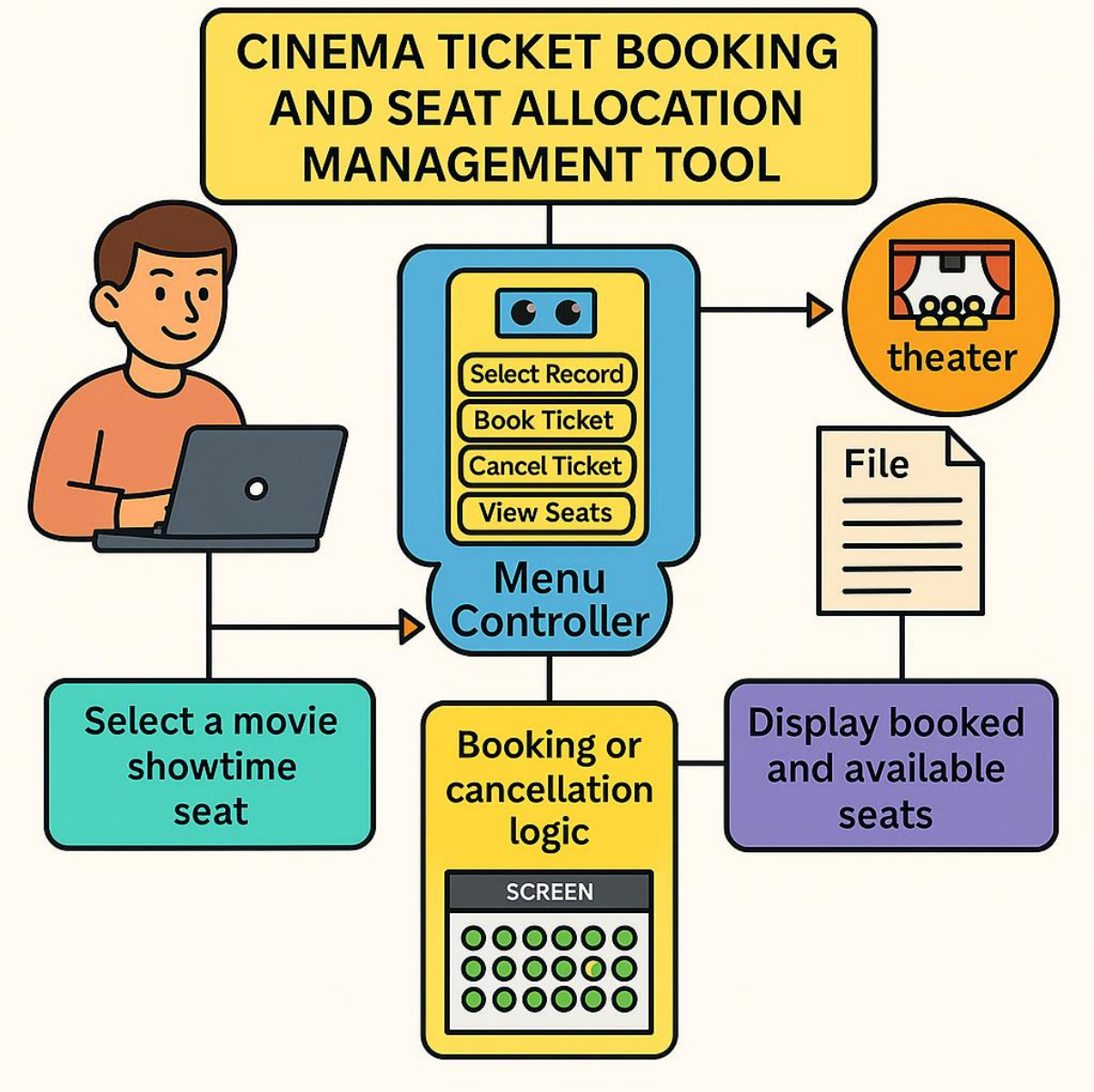
1. Record and update student payments.
2. View pending and completed payments.
3. Save permanent data in *fees.txt*.

Expected Outcome

Students understand how structured data management helps automate financial operations.

15. Cinema Ticket Booking and Seat Allocation Management Tool

Local cinema theaters often face confusion with double seat bookings or missing records. The *Cinema Ticket Booking and Seat Allocation Management Tool* provides a digital solution using C programming. It lets operators manage movie names, show timings, seat numbers, and ticket prices. The system allows customers to book or cancel seats, and automatically marks the availability status in *cinema.txt*. Arrays handle the seating matrix, and conditionals prevent double bookings. The project demonstrates how logical structures and file handling can simulate real-world ticketing systems, ensuring smooth and error-free seat management. It's a perfect example of how simple C programs can replicate commercial applications like PVR or BookMyShow at a basic level.



Why This Project?

- Demonstrates real-life booking and cancellation logic.
- Reinforces matrix handling and validation.
- Encourages real-world software modeling.

Key Features

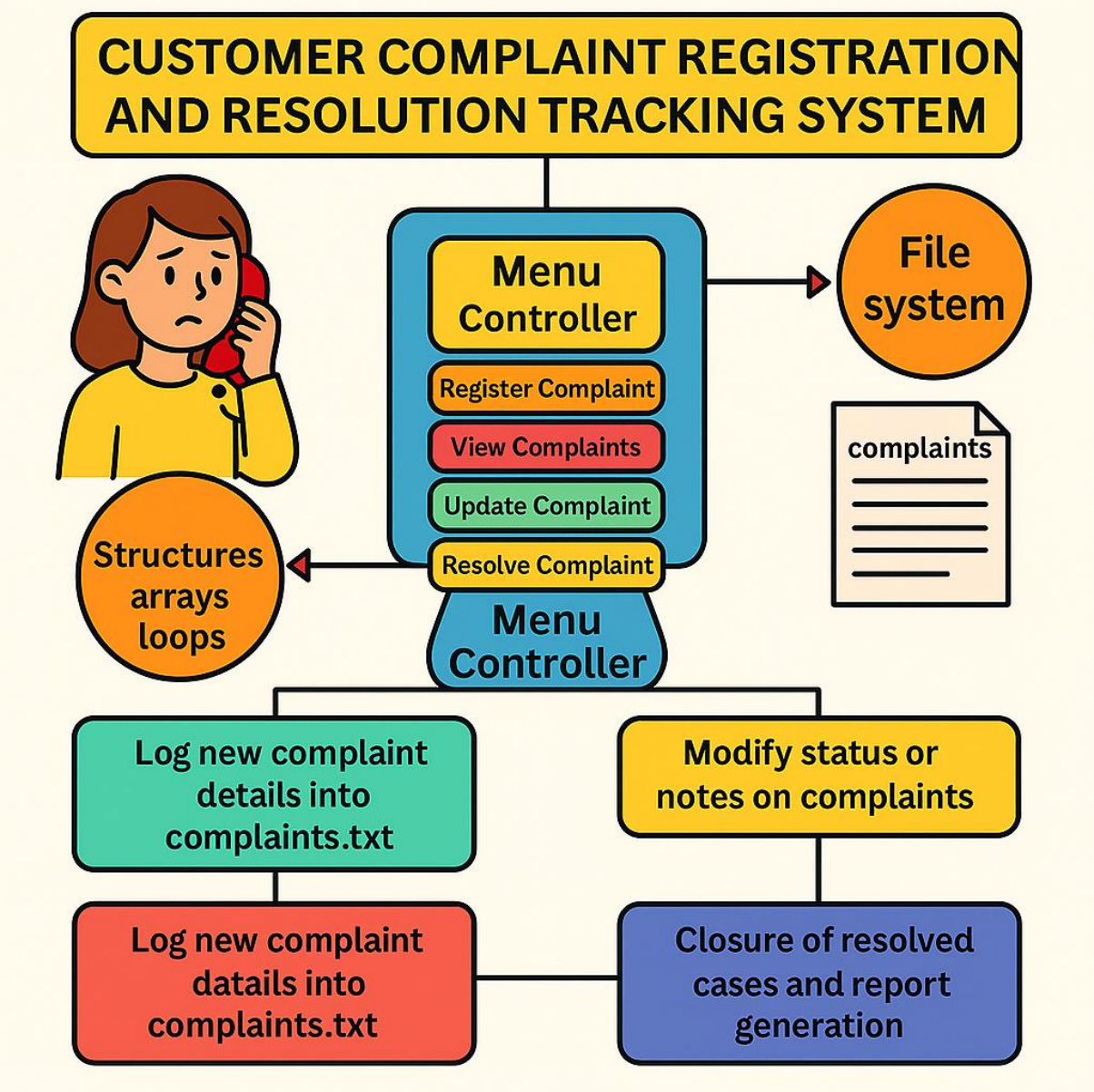
1. Book and cancel seats efficiently.
2. Display available and reserved seats.
3. Save all data in *cinema.txt*.

Expected Outcome

Learners understand booking systems, record automation, and structured data management in real-world contexts.

16. Customer Complaint Registration and Resolution Tracking System

In many service companies—such as telecom and cable networks—customers file complaints that are often misplaced or forgotten. The *Customer Complaint Registration and Resolution Tracking System* brings structure and accountability to this process. Built in C, the system allows executives to log each complaint with an ID, customer name, issue type, and status ("Pending" or "Resolved"). Using loops and conditionals, the program enables searching, updating, and tracking each case efficiently. All records are saved in *complaints.txt*, ensuring that no complaint is lost. This project models the workflow of modern customer service portals, emphasizing the value of data organization and timely feedback. It helps students visualize how digital complaint tracking improves customer satisfaction and operational transparency.



Why This Project?

- Demonstrates case tracking using files and conditionals.
- Builds awareness of workflow-based systems.
- Promotes data persistence and structured reporting.

Key Features

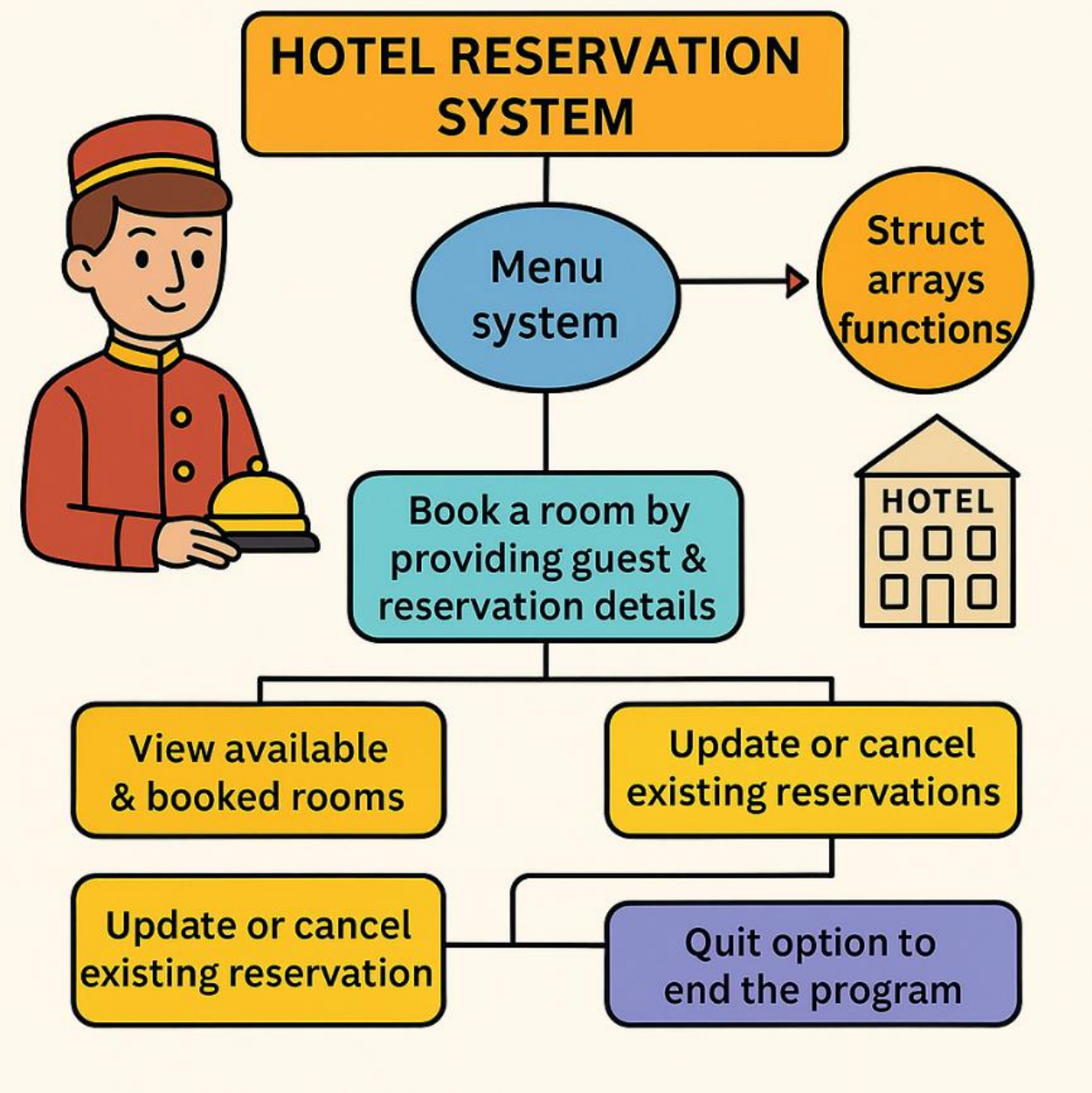
1. Register, update, and search complaints.
2. Generate pending vs. resolved reports.
3. Save all records permanently in *complaints.txt*.

Expected Outcome

Students gain hands-on experience in building practical systems for service-based industries.

17. Hotel Room Reservation and Guest Record Management System

Small hotels and lodges often depend on handwritten registers for reservations, leading to errors and double bookings. The *Hotel Room Reservation and Guest Record Management System* developed in C automates the check-in and booking process. It records guest name, contact number, room type, tariff, and duration of stay using structures. Arrays and functions handle multiple reservations at once, while *hotel.txt* stores all information for future reference. The program can also generate invoices at checkout. This project closely mirrors real hotel management software, teaching how logic and organization reduce manual workload. By integrating data entry, search, and report generation, students understand how automation supports the hospitality industry's daily operations.



Why This Project?

- Simulates hospitality service automation.
- Strengthens concepts of structures and file manipulation.
- Teaches modular and interactive programming.

Key Features

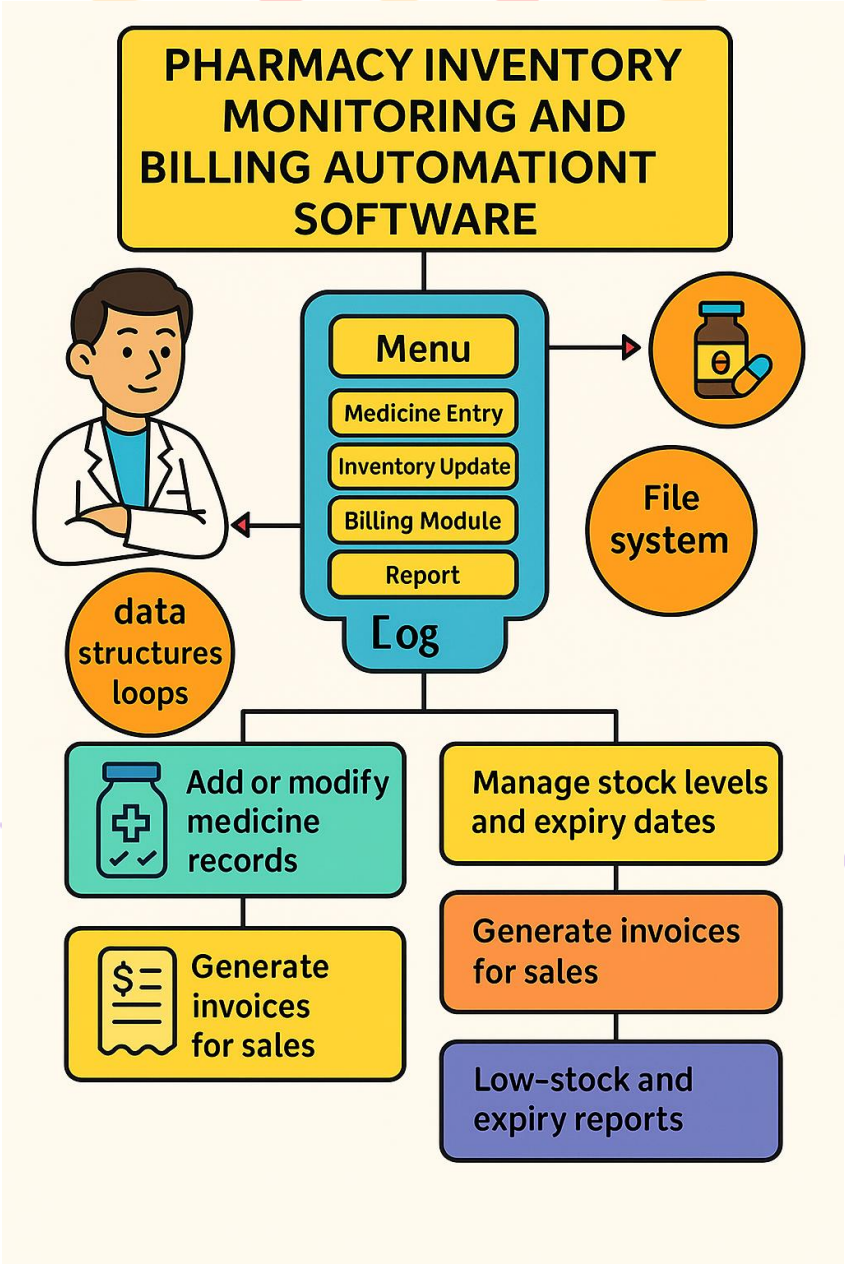
1. Add, cancel, and modify reservations.
2. Generate guest invoices.
3. Store data permanently in *hotel.txt*.

Expected Outcome

Learners understand how structured logic can manage large-scale booking operations seamlessly.

18. Pharmacy Inventory Monitoring and Billing Automation Software

Pharmacies require precise tracking of medicines, prices, and expiry dates. The *Pharmacy Inventory Monitoring and Billing Automation Software* developed in C provides a digital solution for small medical stores. It enables the pharmacist to add new medicines with ID, name, stock quantity, manufacturing, and expiry dates. During billing, the program automatically calculates totals and deducts sold items from inventory. All data is saved in *pharmacy.txt*. This project demonstrates the combination of arithmetic processing and file handling to maintain medical accuracy. It models a simplified version of pharmacy point-of-sale systems and helps students realize how programming can support health-care-related businesses.



Why This Project?

- Focuses on accuracy and real-world data management.
- Integrates billing and inventory concepts.
- Reinforces error handling and arithmetic logic.

Key Features

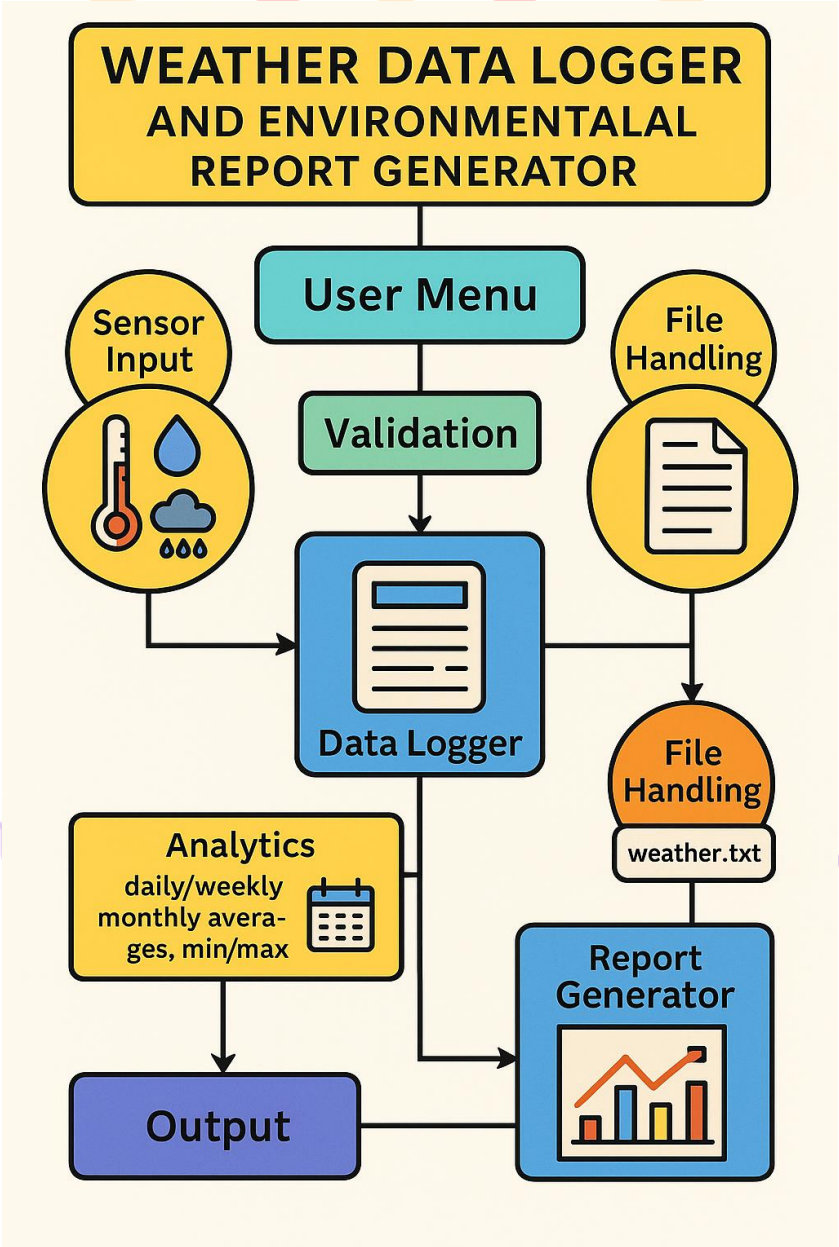
1. Add and update medicine stock.
2. Generate customer bills automatically.
3. Track expiry and store data in *pharmacy.txt*.

Expected Outcome

Students understand integrated stock-billing workflows and how software improves accuracy in pharmacy operations.

19. Weather Data Logger and Environmental Report Generator in C

Meteorological departments rely on continuous weather readings for temperature, humidity, and rainfall. The *Weather Data Logger and Report Generator* written in C simulates this process by allowing users to enter daily environmental readings. The program calculates weekly and monthly averages and saves each entry in *weather.txt*. Structures and arrays organize multiple data points, while loops enable repetitive data entry and analysis. This project demonstrates how a simple C console application can serve as a foundation for environmental monitoring systems, similar to those used in rural weather centers or school laboratories. It merges computation with awareness, showing students how code contributes to climate analysis and data reporting.



Why This Project?

- Illustrates numeric data recording and averaging.
- Encourages environmental applications of programming.
- Strengthens file handling and calculation logic.

Key Features

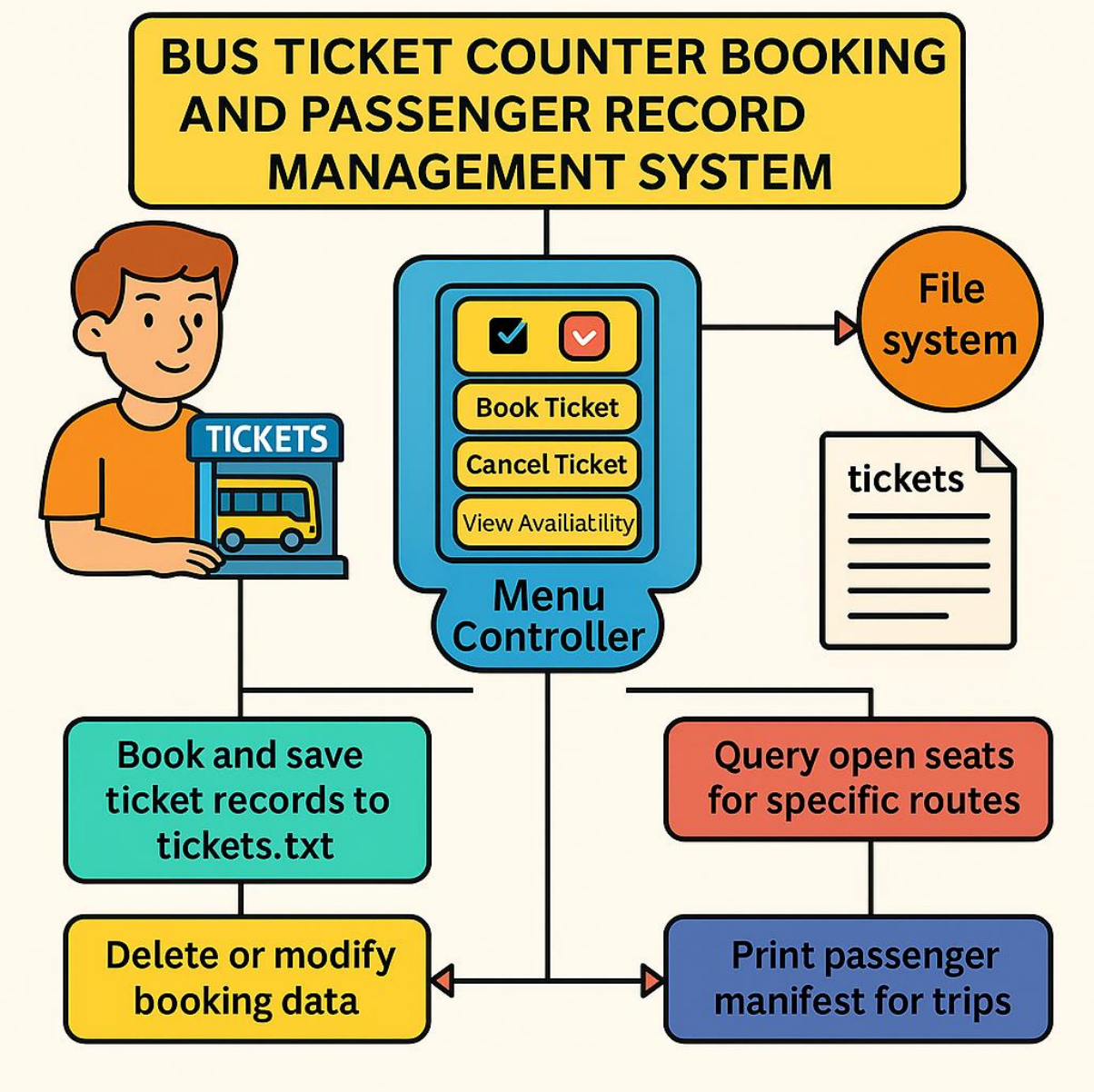
1. Record daily temperature and rainfall.
2. Compute and display averages.
3. Store all readings in *weather.txt*.

Expected Outcome

Students learn how scientific observations can be automated through code and structured data storage.

20. Bus Ticket Counter Booking and Passenger Record Management System

In many intercity bus depots, staff still manage bookings manually, leading to confusion during peak seasons. The *Bus Ticket Counter Booking and Passenger Record Management System* digitizes the entire process using C programming. It allows the operator to book, cancel, or modify tickets while maintaining data such as bus number, passenger name, destination, seat number, and fare. Arrays manage the seat matrix efficiently, ensuring no duplicate bookings. All information is written to *bus.txt* for long-term use. The system generates summary reports at the end of the day, showing total passengers and revenue collected. It reflects how transport companies automate their booking counters, allowing students to experience the logic behind real-world scheduling and passenger management.



Why This Project?

- Demonstrates real transport management processes.
- Strengthens conditional and loop logic.
- Reinforces practical file-handling applications.

Key Features

1. Book and cancel passenger tickets.
2. Check seat availability.
3. Save trip details in *bus.txt*.

Expected Outcome

Learners grasp how transportation systems can be designed and controlled through programming.