

Ahsanullah University of Science and Technology (AUST)

Department of Electrical and Electronic Engineering

Course No.: EEE 1110

Course Title: Programming Language Laboratory

****Restaurant Table Reservation System****

ID: 20230205183

Name: MD Fuad Hasan

Section: C2

For the students of the

Department of Electrical and Electronic Engineering

1st Year, 1st Semester

### ****Overview****

The Restaurant Table Reservation System is designed to manage table bookings efficiently, providing functionality for booking tables, viewing booking details, checking table status, searching for customers, and canceling or complete reservations.

### ****Objective****

The primary objective of the system is management of table reservations in a restaurant setting. Key functionalities include:

1. **Booking Tables**: Allow customers to reserve tables.
2. **Viewing Booking Information**: Display existing reservations and customer details.
3. **Checking Table Status**: Show the availability status of all tables.
4. **Searching for Customers**: Locate specific customer bookings.
5. **Canceling/ Complete Reservations**: Remove or modify existing bookings.

### ****Design and Implementation****

### Booking ****(Structure)****

Represents individual bookings in the system. Encapsulates all necessary details of a booking in a single structure for easy management and manipulation.

* int bookingNum: Unique identifier for each booking.
* string name: Name of the customer who made the reservation.
* int age: Age of the customer.
* string phoneNum: Contact number of the customer.
* int tableNum: Table number reserved by the customer.

**Table (Structure)**

Represents the status of each table. Provides a straightforward way to track the reservation status of tables in the restaurant.

* bool reserved: Indicates whether the table is reserved.
* string status: Status of the table, either "Available" or "No available".

**Admin (Class)**

Manages all administrative functionalities, including logging in, booking tables, viewing statuses, and handling customer information. Centralizes all administrative operations, making it easier to maintain and manage the system.

* string password: Admin password for authentication.
* void viewCustomerInfo(): Displays all customer bookings.
* void bookTable(): Allows table reservations.
* void viewTableStatus(): Shows the current status of tables.
* void searchCustomer(): Searches for customer bookings by name.
* void cancelBooking(): Cancels a reservation.
* void logout(): Logs out the admin.
* void menu(): Displays the main menu and handles user input.
* vector<Booking> loadBookings(): Loads booking information from a file into a vector.

### ****Functions****

**viewCustomerInfo()**

Reads booking information from the file, sorts it by table number, and displays it. Uses vector<Booking> for managing and sorting bookings, and outputs them in a formatted table.

**bookTable()**

Allows the admin to reserve a table for a customer by updating the booking file. Checks table availability, handles input validation for phone numbers, and appends new bookings to the file.

**viewTableStatus()**

Shows the reservation status of each table in the restaurant. Updates table statuses based on current bookings and displays them to the admin.

**searchCustomer()**

Searches for and displays bookings based on the customer's name. Iterates through the bookings to find matches and displays relevant details.

**cancelBooking()**

Removes a booking from the system based on the booking number. Reads existing bookings, writes all but the specified booking to a temporary file, and then replaces the old file with the updated one.

**logout()**

Ends the admin session and returns to the login screen. Clears the screen and prompts for login credentials again.

**menu()**

Displays the main menu options and handles user selections. Uses a switch-case structure to call appropriate methods based on user input.

**login()**

Authenticates the admin using a password. Compares entered password with the stored password and grants access to the menu if correct.

### ****Problems During Implementation****

**File Management Issues**: Handling file operations for reading and writing can be error-prone, particularly with concurrent access and ensuring data consistency.

**Input Validation**: Ensuring that user inputs (like phone numbers) are valid and conform to expected formats can be challenging.

**Data Sorting and Display**: Efficiently sorting and displaying data while maintaining readability required careful use of data structures and formatting.

### ****Future Improvements****

**User Interface**: Transitioning from a console-based interface to a graphical user interface (GUI) could improve usability.

**Error Handling**: Adding more robust error handling for file operations and user inputs.

**Security Enhancements**: Implementing secure password management and encryption for sensitive data.

**Concurrency**: Managing concurrent access to the file if multiple instances of the application are running simultaneously.

### ****Conclusion****

The Restaurant Table Reservation System provides a functional and efficient solution for managing table reservations. The design focuses on simplicity and effectiveness, using basic file operations and data structures to handle booking details. Future enhancements can further improve the system's robustness, user experience, and security.

This report outlines the rationale behind the design decisions, implementation details, and potential areas for improvement, providing a comprehensive overview of the system's capabilities and future direction.