# PROJECT REPORT: XYZ BUS MANAGEMENT SYSTEM

****

## **SUBMITTED BY:**

MUHAMMAD AHMAD 24K-3078

HAMZA SALAHUDDIN 24K-3065

MUHAMMAD HASHIR 24K-3039

## **SUBMITTED TO:**

MISS RAVIA EJAZ

# **INTRODUCTION:**

The XYZ Public Transport System project is a C-based program designed to manage ticket booking, passenger tracking, and loyalty profiles for a public transport system. The system is intended to handle up to 10 buses with a maximum capacity of 40 passengers each, and provides features such as discount calculation, peak hour surcharges, and user profiles.

# **OBJECTIVES:**

* To provide a streamlined ticket booking system for passengers.
* To implement discounts and peak hour surcharges for accurate fare calculation.
* To maintain daily passenger records and loyalty profiles.
* To ensure administrative functionalities such as record viewing and resetting.

# **METHODOLOGY:**

The system is built using the C programming language, utilizing various modules for specific tasks. The key functionalities are divided into passenger ticket booking, discount application, and administrative controls. Data persistence is managed through text files for storing passenger records and loyalty profiles.

# **IMPLEMENTATION DETAILS:**

## 1. Core Modules

* **Passenger Ticket Booking**:

Handles booking process including seat allocation, fare calculation, and storing data.

* **Discount and Surcharge Calculation**:

Applies discounts based on payment method, height, and number of stops. Adds a peak hour surcharge if applicable.

* **Admin Panel**:

Provides options for viewing daily records, resetting them, and viewing loyalty profiles.

## 2. Data Management

* **Passenger Records**:

Stored in passenger\_records.txt for daily transaction logs.

* **Loyalty Profiles**:

Stored in loyalty\_data.txt to maintain passenger history and fare data.

## 3. Algorithms and Logic

* **Seat Allocation**:

Iterates through buses and seats to find the first vacant seat.

* **Fare Calculation**:

Base fare is $20 with an additional $10 per stop after the first.

* **Discount Application**:  
   Applies cumulative discounts of 3% for card payments, trips longer than two stops, and passengers under a height of 3 feet.
* **Peak Hour Surcharge**:  
   Adds a 5% surcharge during peak hours (5 PM - 8 PM).

# **CODE EXPLANATION:**

## 1. Main Menu:

* The program starts by displaying the main menu, which includes options for "Start Boarding", "Admin Panel", and "Exit".
* The user's choice is obtained, and the corresponding functionality is executed.

## 2. Process Boarding:

* This subgraph handles the boarding process at each stop.
* It displays the current stop, prompts the user to book a ticket, and calls the  bookTicket function if the user chooses to do so.
* After finishing the boarding process at the current stop, it moves to the next stop.

## 3. Book Ticket:

* This subgraph is responsible for the ticket booking process.
* It generates a unique customer ID, prompts the user for their details (name, age, height, payment method), and gets the ending stop.
* It calculates the fare based on the number of stops, applies discounts (based on height, payment method, and number of stops), and applies peak hour charges if applicable.
* It assigns a seat and bus, generates the booking time, saves the passenger details to a file, updates the user's loyalty profile, and displays the booking details.

## 4. Admin Panel:

* This subgraph handles the admin functionality.
* It first verifies the admin password, and if the password is correct, it displays the admin menu.
* The admin can choose to view daily records, reset the records, or view loyalty profiles.
* The corresponding functionality is executed based on the admin's choice.

## 5. Utilities:

* This subgraph contains the implementation of utility functions used throughout the program.
* applyDiscount: Applies discounts based on the payment method, number of stops, and passenger height.
* applyPeakHourCharge: Applies a peak hour surcharge if the booking time is between 5 PM and 8 PM.
* updateUserProfile: Updates the user's loyalty profile by incrementing the number of trips, updating the total distance, and updating the total fare paid.
* saveUserProfile: Saves the user's loyalty profile to a file.

# **FLOWCHARTS:**

## Main Function

BACK TO DISPLAY MENU

BOARD PASSENGERS AT STOPS

ADMIN PANNEL

EXIT SYSTEM

GET USER CHOICE

DISPLAY MENU

CHOICE 2

CHOICE 1

CHOICE 3

## Book Tickets

NO

YES

DISPLAY BOOKING DETAILS

UPDATE FILES

ASSIGN SEAT

APPLY DISCOUNTS

EXIT STOP > CURRENT STOP?

GENERATE CUSTOMER ID

GET CUSTOMER DETAILS

## Admin Panel

INCORRECT

CORRECT

CHOICE 1

CHOICE 3

CHOICE 2

BACK TO MAIN FUNCTON

DISPLAY ERROR

DISPLAY DAILY RECORDS

DISPLAY LOYALTY RECORDS

CLEAR RECORD

CHOOSE OPTION

ADMIN OPTIONS

VALIDATE PASSWORD

GET PASSWORD

## Processing Passengers At Stops

N

Y

MOVE TO NEXT STOP

BOOK TICKET FUNCTION

USER CHOICE

PROMPT USER INPUT TO BOOK TICKET

DISPLAY CURRENT STOP

# **PSEUDO CODE:**

## Main Function

1. Initialize random seed
2. While True:
3. Display Menu
4. Get User Choice
5. If Choice == 1:
   1. For each stop from 0 to MAX\_STOPS-1:
   2. Process Passengers At Stop (stopIndex)
6. Else If Choice == 2:
   1. Admin Panel
7. Else If Choice == 3:
   1. Print "Exiting..."
   2. Break
8. Else:
   1. Print "Invalid choice."

## Book Tickets

1. Create a Passenger object
2. Generate Customer ID for the passenger
3. Print "Your new customer ID is: [customerID]"
4. Get New Customer Details for the passenger
5. Repeat:
6. Print "Select the ending stop (greater than [stopIndex + 1] to [MAX\_STOPS]): "
7. Get endChoice
8. If endChoice <= stopIndex:
   1. Print "Invalid stop selection. The ending stop must be greater than the current stop."
9. Until endChoice > stopIndex
10. Set passenger.endingStop to stops[endChoice - 1]
11. Calculate numberOfStops as endChoice - stopIndex
12. Calculate passenger.distance as numberOfStops \* ADDITIONAL\_STOP\_FARE
13. Calculate passenger.fare as BASE\_FARE + (numberOfStops - 1) \* ADDITIONAL\_STOP\_FARE
14. Apply Discount to passenger.fare based on height, payment method, and number of stops
15. Apply Peak Hour Charge to passenger.fare
16. For each bus i from 0 to MAX\_BUSES-1:
17. For each seat j from 0 to SEATS\_PER\_BUS-1:
    1. If seats[i][j] is vacant:
    2. Assign passenger.busNumber to i + 1
    3. Assign passenger.seatNumber to j + 1
    4. Mark seat as booked (seats[i][j] = 1)
    5. Goto SeatAssigned
18. If all seats are full:
19. Print "All seats are full!"
20. Return
21. SeatAssigned:
22. Get current time and assign to passenger.bookingTime
23. Save Passenger to File
24. Update Loyalty Profile of the passenger
25. Display Booking Details and Discount Information

## Admin Panel

1. Print "Enter admin password: "
2. Get password
3. If password is valid:
4. Print "Admin Panel"
5. Print "1. View Daily Records"
6. Print "2. Reset Records"
7. Print "3. View Loyalty Profiles"
8. Print "Enter your choice: "
9. Get choice
10. If choice == 1:
    1. View Daily Records
11. Else If choice == 2:
    1. Reset Daily Records
12. Else If choice == 3:
    1. View Loyalty Profiles
13. Else:
    1. Print "Invalid choice."
14. Else:
15. Print "Invalid password!"

## Process Passengers At Stops

1. Repeat:
2. Print "Now boarding passengers at [current stop]"
3. Print "Do you want to book a ticket for this stop? (y/n): "
4. Get choice
5. If choice == 'y' or choice == 'Y':
   1. Book Ticket (stopIndex)
6. Else:
   1. Print "Finished boarding at [current stop]. Moving to the next stop."
7. Until choice != 'y' and choice != 'Y'

## Auxiliary Functions

1. Save to File(passenger):
   1. Open file in append mode
   2. Write passenger details to file
   3. Close file
2. Generate Customer ID:
   1. Return a random number between CUSTOMER\_ID\_MIN and CUSTOMER\_ID\_MAX
3. Get New Customer Details(passenger):
   1. Print "Enter your name: "
   2. Get name and assign to passenger
   3. Print "Enter your age: "
   4. Get age and assign to passenger
   5. Print "Enter your height (in feet): "
   6. Get height and assign to passenger
   7. Print "Enter your payment method (cash/card): "
   8. Get payment method and assign to passenger
4. Apply Discount(fare, height, paymentMethod, numberOfStops, discountDetails):
   1. Initialize discountApplied to 0
   2. If paymentMethod == "card":
      1. Apply 3% discount to fare
      2. Append "Card payment (3%)" to discountDetails
      3. Set discountApplied to 1
   3. If numberOfStops > 2:
      1. Apply 3% discount to fare
      2. Append "More than 2 stops (3%)" to discountDetails if discountApplied is set
      3. Set discountApplied to 1
   4. If height < DISCOUNT\_HEIGHT\_THRESHOLD:
      1. Apply 3% discount to fare
      2. Append "Height less than 3 feet (3%)" to discountDetails if discountApplied is set
   5. Return fare
5. Apply Peak Hour Charge(fare):
   1. Get current time
   2. If current time is within peak hours:
      1. Apply 5% peak hour surcharge to fare
      2. Print "Peak hour charge applied (5%)"
   3. Return fare
6. Update User Profile(user, distance, fare):
   1. Increment tripsCompleted of user
   2. Add distance to totalDistance of user
   3. Add fare to totalFarePaid of user
   4. Save User Profile
7. Save User Profile(user):
   1. Open file in append mode
   2. Write user profile data to file
   3. Close file
8. View Daily Records:
   1. Open file in read mode
   2. If file does not exist:
      1. Print "No records found."
      2. Return
   3. Read and print each line of the file
   4. Close file
9. Reset Daily Records:
   1. Open file in write mode to clear content
   2. Close file
   3. Print "Daily records reset successfully."
10. View Loyalty Profiles:
    1. Open file in read mode
    2. If file does not exist:
       1. Print "No loyalty profiles found."
       2. Return
    3. Read and print each line of the file
    4. Close file

# **RESULTS:**

The XYZ Public Transport System effectively manages passenger ticket booking, applies discounts and surcharges, and tracks loyalty profiles. Daily records and loyalty data are successfully stored in respective files, ensuring data persistence.

## Sample Outputs

A sample booking process includes:

* Passenger enters personal details and selects an ending stop.
* Discounts and peak hour surcharges are applied.
* The system assigns a bus and seat, and saves the booking.
* Admin can view and reset records or view loyalty profiles.

# **CONCLUSION:**

The XYZ Public Transport System is a comprehensive solution for managing public transport ticketing operations. The system is scalable, user-friendly, and maintains essential data records, making it a robust choice for transport management.

# REFERENCES:

* Geeks for Geeks
* Code with Harry
* W3schools

# LINKS:

**GITHUB:**

<https://github.com/Muhammad-Ahmad135/PF-FINAL-PROJECT>