CAPSTONE PROJECT

DATA SCIENCE PROJECT

presented by:
R.Karthick-Btech.in . chemical engineering
college name: Algappha college of technology Chennai (Anna University)

OUTLINE

- **Problem Statement**

- **Proposed System/Solution**

- **System Development Approach**

- **Algorithm & Deployment**

- **Result**

- **Conclusion**

- **Future Scope**

- **References**

PROBLEM STATEMENT

Python Program for Railway Reservation System

PROPOSED SOLUTION

To develop a Python program for railway ticket booking, you'll need to design and implement a system that handles various aspects such as train schedules, seat availability, user authentication, booking transactions, and more. Below are proposed solutions covering key functionalities and considerations for building a railway ticket booking program in Python:

1. **Object-Oriented Design (OOD):** Use classes and objects to model entities like Train, Ticket, User, and Booking. This facilitates a structured approach to representing railway system components.

2. **User Authentication and Management:** Implement user authentication functionalities to allow users to create accounts, log in, and manage their profiles.

3. Train and Seat Management Solution: Manage train schedules, seat availability, and bookings within the booking system.

4. Booking Transactions: Implement logic to handle booking transactions, including seat reservations and ticket issuance.

5. Data Persistence and Storage: Implement data storage using file handling or databases (e.g., SQLite, PostgreSQL) to persist train schedules, user information, and booking records.

SYSTEM APPROACH

1. User Interface:
- **Frontend Interface:** Users interact with the system through a graphical user interface (GUI) or command-line interface (CLI).
- **Input Validation:** Validate user inputs to ensure correctness and prevent invalid data entry.

2. Data Management:
- **Trains Data:**
  ○ Maintain a database or data structure to store information about trains, including train number, name, available seats, departure time, and possibly other attributes like class, fare, etc.
  ○ Store this data in a format that allows for efficient retrieval and modification.
- **Bookings Data:**
  ○ Store information about bookings, including booking ID, train number, number of seats booked, and passenger details (name, age, contact).
  ○ Each booking should be linked to a specific train and have a unique identifier.

Functionalities:

- **Display Available Trains:**
  - Retrieve information about available trains and display them to the user.
  - Include details like train number, name, available seats, departure time, etc.
- **Book Tickets:**
  - Allow users to book tickets for a selected train.
  - Collect necessary details like train number, number of seats, passenger information.
  - Update the available seats for the selected train after booking.
- **Cancel Tickets:**
  - Enable users to cancel their booked tickets by providing the booking ID.
  - Update the available seats for the respective train after cancellation.
- **View Booking Details:**
  - Allow users to view details of their booked tickets by providing the booking ID.
  - Display booking details including train number, number of seats booked, passenger details, etc.

4. Business Logic:

- **Validation:**
  - Validate user inputs to ensure they are within acceptable ranges and formats.
  - Check for seat availability before booking.
  - Handle edge cases and error conditions gracefully.
- **Concurrency Control:**
  - Implement measures to handle concurrent access to the same data, preventing race conditions and ensuring data integrity.
  - Use locks or other synchronization mechanisms if necessary.

5. Security:

- **User Authentication:**
  - Implement user authentication to ensure only authorized users can access certain functionalities (e.g., booking, cancellation).
  - Protect sensitive data like user passwords using encryption.
- **Data Privacy:**
  - Ensure that passenger details are stored securely and access is restricted to authorized personnel only.

6. Administration:
- **Admin Interface:**
  - Provide an interface for administrators to manage trains, bookings, and user accounts.
  - Allow admins to add/remove trains, modify seat availability, view booking history, etc.

7. Error Handling and Logging:
- **Error Handling:**
  - Handle errors gracefully and provide informative error messages to users.
  - Log errors and exceptions for debugging and monitoring purposes.

ALGORITHM & DEPLOYMENT

Algorithm:

1. Initialization:

- Initialize the RailwayReservationSystem class.

- Initialize train data and booking data (either from a file/database or hardcoded).

  2. Display Menu:

- Display a menu with options: Display Trains, Book Ticket, Cancel Ticket, View Booking Details, and Exit.

  3. Display Trains:

- Iterate through the train data and display train details (train number, name, departure time, and available seats).

4. Book Ticket:

- Prompt the user to enter the train number, number of seats, and passenger details.

- Check if the train exists and if there are enough available seats.

- If valid, decrement the available seats, generate a booking ID, and store booking details.

- Display a success message with the booking ID.

5. Cancel Ticket:

- Prompt the user to enter the booking ID to cancel.

- Check if the booking ID exists.

- If valid, increment the available seats, remove the booking from the booking data.

- Display a success message.

6. View Booking Details:

- Prompt the user to enter the booking ID.
- Display the details of the booking if the ID exists.

7. Exit:

- Exit the program.

Deployment:

1. Local Deployment:

- **Run Locally:** Run the Python program on a local machine.

- **Installation:** Ensure Python is installed. Execute the program in a Python IDE or command line.

- **Advantages:** Simple setup, suitable for development and testing.

- **Disadvantages:** Not accessible remotely, requires installation on each machine.

2. Web Application:

- **Web Framework:** Use Python web frameworks like Flask or Django.

- **Deployment Platform:** Deploy the web application on platforms like Heroku, AWS, or DigitalOcean.

- **Advantages:** Accessible from any device with an internet connection, scalable, and easier maintenance.

- **Disadvantages:** More complex setup, requires web development skills.

3. Cloud Deployment:

- **Serverless Computing:** Utilize serverless platforms like AWS Lambda or Google Cloud Functions.
- **Containerization:** Use Docker containers and deploy on container orchestration platforms like Kubernetes.
- **Advantages:** Scalable, cost-effective, and managed infrastructure.
- **Disadvantages:** Learning curve for setting up serverless or containerized environments.

4. Desktop Application:

- **GUI Framework:** Use Python GUI frameworks like Tkinter, PyQt, or wxPython.
- **Deployment:** Package the application using tools like PyInstaller or cx_Freeze.
- **Advantages:** No need for internet connection, easy distribution.
- **Disadvantages:** Platform-dependent, less accessible than web applications.

RESULT

```
dibya@dibya-HP-240-G5-Notebook-PC ~/workspace/railway-reservation $ python3 reserv.py
--------------------------------------------Welcome to Railway Reservation Portal--------------------------------------------
--------------------------------------------------------------------------------------------------------------------------
Choose one of the following option:-
1.Book Ticket
2.Cancel Ticket
3.Check PNR
4.Check seat availibity
5.Create new account
6.Check previous bookings
7.Login
8.Exit
Enter your option :- 1
Enter the User ID:- 1212
Enter your password:- dibyadas

Welcome  dibya das  !

Enter the source station:- ctc
Enter the destination station:- kgp
Train Name :-  odisha   Number  12345   Day of Travel :-  Wed
Enter the train number :- 12345
No. of seats available in 1AC :- 30
No. of seats available in 2AC :- 23
No. of seats available in SL :- 43
Enter the coach :- sl
Enter the number of tickets :- 3
You have to pay :-  702
Confirm? (y/n) :-y
Booking Successful!
```

CONCLUSION

The Railway Reservation System is a crucial system that facilitates the booking and management of train tickets. In this system, we have designed a Python program to handle basic functionalities like booking tickets, canceling tickets, and viewing train details.

Key Points:

1. **Functionality:** Users can view available trains, book tickets for a specific train, cancel booked tickets, and view booking details.

2. **Data Management:** Train data and booking data are stored using dictionaries and lists.

3. **User Interface:** The system provides a simple menu-driven interface for users to interact with.

4. Error Handling: Input validation and error messages are provided to ensure smooth user experience.

5. System Design: The system is designed with clear components: user interface, data management, functionality, error handling, and control flow.

6. Optional Features: The system can be extended with features like user authentication, admin interface, and data persistence.

FUTURE SCOPE

1. Mobile Ticketing and NFC Technology

2. AI-Powered Customer Service

3. Dynamic Pricing and Revenue Management

   4. Predictive Maintenance

   5. Biometric Identification and Security

   6. Integration with Public Transport

   7. Augmented Reality (AR) for Navigation

THANK YOU