A Project Report

On

Ticket Reservation System

Submitted by

Tooba Ali Kazmi 23-08

Farwa Zainab 23-07

Momana Batool 23-41

Fariha Tayyab 23-13

Fiza Ashiq 23-24

Submitted by

Faisal Hafeez

 University Of Layyah

Acknowledgment:

We would like to thank our course instructor, Mr. Faisal Hafeez, for his continuous guidance and encouragement throughout this project. We also extend our gratitude to our peers for their support and feedback.

Summary:

The Ticket Reservation System is a Python-based application designed to simplify ticket booking, management, and cancelation. The project demonstrates the use of data structures such as arrays, linked lists, and queues to handle reservations efficiently. It provides a command-line interface for interaction, ensuring a user-friendly experience. This project effectively integrates programming concepts like error handling, file management, and modular design. Key features include booking tickets, viewing reservations, and canceling bookings. The system is scalable and can be extended for real-world use in transportation or event management.

Introduction:

A Ticket Reservation System is an essential application used in various industries, including transport, events, and entertainment. The purpose of this project is to develop a Python-based system that efficiently manages ticket reservations while ensuring user satisfaction. This project focuses on creating a robust backend structure, enabling smooth functionality and data integrity.

Objectives:

Develop a system for efficient ticket booking and management.

Implement a robust structure for handling user data securely.

Provide a seamless user experience through a command-line interface.

Demonstrate the application of data structures in solving real-world problems.

Tools and Technologies:

Programming Language: Python

IDE: PyCharm, Visual Studio Code

Data Structures: Arrays, Linked Lists, Queues

Version Control: Git

Testing Tool: Python unittest module

Methodology:

1. Requirement Gathering: Identified features such as booking, viewing, and canceling tickets.
2. Design Phase: Designed the system architecture using UML diagrams and flowcharts.
3. Implementation: Coded the core modules for reservation, management, and error handling.
4. Testing: Validated the system functionality using test cases.
5. Deployment: Created a Python-based CLI application for interaction.

Implementation:

The system consists of the following modules:

1. User Module:

Registers users and authenticates them. import tkinter as tk

from tkinter import messagebox

import time

from datetime import datetime

# Function to add a reservation to the queue

def add\_reservation():

name = entry\_name.get()

if name.strip():

timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")

ticket\_queue.append((name, timestamp))

entry\_name.delete(0, tk.END)

update\_queue\_display()

messagebox.showinfo("Success", f"Reservation for '{name}' added!")

else:

messagebox.showwarning("Input Error", "Please enter a name!")

1. Booking Module:

Allows users to book tickets by providing necessary details like name, date, and seat number.

Uses arrays to store and retrieve bookings efficiently.

# Function to process the next reservation

def process\_reservation():

if ticket\_queue:

processed\_name, \_ = ticket\_queue.pop(0)

update\_queue\_display()

messagebox.showinfo("Processed", f"Reservation for '{processed\_name}' processed!")

else:

messagebox.showwarning("Queue Empty", "No reservations to process!")

1. Cancelation Module:

Enables users to cancel reservations by referencing their booking ID.

Implements linked lists to dynamically update bookings.

# Function to cancel a specific reservation

def cancel\_reservation():

name = entry\_name.get()

for reservation in ticket\_queue:

if reservation[0] == name:

ticket\_queue.remove(reservation)

entry\_name.delete(0, tk.END)

update\_queue\_display()

messagebox.showinfo("Cancelled", f"Reservation for '{name}' cancelled!")

return

messagebox.showwarning("Not Found", f"Reservation for '{name}' not found!")

1. View Module:

Displays all booked tickets in a user-friendly format.

Utilizes queues to ensure sequential access.

# Listbox to display the queue

label\_queue = tk.Label(root, text="Current Reservations:")

label\_queue.pack(pady=5)

queue\_display = tk.Listbox(root, width=50, height=10)

queue\_display.pack(pady=5)

# Run the GUI

root.mainloop()

1. Results:

* The Ticket Reservation System successfully achieved the following:
* Efficiently handled booking, cancelation, and viewing operations.
* Provided accurate and reliable data storage.
* Demonstrated the practical application of data structures in solving real-world problems.
* Ensured scalability for future enhancements, such as integrating a graphical .



 

 

