



“First Come First Served Disk Scheduling for Ticket Reservation”

GROUP MEMBERS:

ABDUL HASEEB {192211497}

G. K. Ranjith {192221120}


SHAIK JAMEEL AHAMMAD {192210638}

CSA0488 – Operating
System

Faculty Name:
Hemavathi R



Problem Statement:

- Design a clean and intuitive dashboard that allows users to easily navigate through the reservation process.
 - Provide a search functionality for users to check table availability based on criteria such as date, time, party size, and dining preferences.
 - Implement an interactive calendar for users to choose their preferred reservation date and time visually.
 - Ensure real-time updates on table availability to avoid conflicts and provide accurate information to users.
 - Allow users to create profiles to save their preferences, view reservation history, and manage upcoming reservations.
 - Provide filters for users to customize their search based on criteria like cuisine type, location, or special amenities.
 - Display a visual representation of the restaurant layout with available tables marked, helping users choose the most suitable seating.
- 

Proposed Design Work:

Identifying the key Components:

- Search and Reservation Form: Allows users to input ticket details, such as destination, seat preference, and departure time.
- Reservation Service: Manages the FCFS disk scheduling algorithm for ticket reservation.
- Disk Allocation Service: Allocates disk blocks for storing ticket information in a first-come-first-served manner.
- Database Integration: Communicates with a database to store and retrieve reservation data.
- Ticket Reservation Database: Stores information about reserved tickets, including destination, seat number, and departure time.

Functionality

- Initialize the disk with a predefined number of blocks.
- Initialize the reservation queue to manage incoming ticket reservation requests.
- Receive reservation requests containing ticket details (e.g., destination, seat preference, departure time).
- Enqueue the reservation request in the FCFS queue.
- Allocate disk blocks for storing ticket information based on the FCFS scheduling algorithm.
- Retrieve the next available block from the FCFS queue.
- Store the ticket information in the allocated disk block.
- Include details such as destination, seat number, and departure time.
- Generate a confirmation message containing the details of the reserved ticket.
- Include information such as reservation ID, destination, seat number, and departure time.
- Send an instant confirmation notification to the user who made the reservation.

Architectural Design

1. User Interface (UI):

- Frontend application developed using a modern web framework (e.g., React, Angular, or Vue.js).
- Responsible for receiving reservation requests, displaying confirmation messages, and allowing users to interact with the system.

2. Backend Services:

- Backend server developed using a server-side language (e.g., Node.js, Python, or Java).
- Implements the FCFS disk scheduling algorithm, reservation handling, and disk block allocation.

3. Database:

- Relational database (e.g., PostgreSQL, MySQL) to store reservation data.
- Tables for storing reservation details, including destination, seat number, departure time, and allocated disk block information.

4. APIs (Application Programming Interfaces):

- RESTful APIs to facilitate communication between the frontend and backend.
- Separate APIs for reservation requests, confirmation generation, cancellation, and retrieval of reservation information.

5. Disk Scheduling Module:

- Manages the FCFS scheduling logic for allocating and deallocating disk blocks.

UI Design:

Layout Design:

The reservation form, where users input their ticket details, should be prominently placed on the homepage or a dedicated reservation page.

By having the reservation form easily accessible, users can quickly initiate the reservation process without unnecessary navigation. Place relevant fields such as destination, seat preference, and departure time in a clear and logically organized layout. This intuitive placement improves user experience and encourages users to interact with the reservation system effortlessly.

Feasible elements used:

Creating a web-based UI allows users to access the reservation system from various devices with internet connectivity, providing flexibility and broad accessibility.

Implementing a responsive design ensures that the user interface adapts seamlessly to different screen sizes, enhancing user experience across desktops, tablets, and smartphones.

Using a server-side language like Node.js, Python (Django/Flask), or Java ensures the development of robust backend services for processing reservation requests and managing disk scheduling.

Implementing the FCFS disk scheduling algorithm provides a straightforward and fair method for allocating disk blocks, making it feasible for a reservation system with a predictable order of reservations.

Feasibility Positioning :

The FCFS algorithm is a simple and straightforward disk scheduling technique. Its feasibility is high for a ticket reservation system where a fair and predictable order of reservations is desired. The algorithm's positioning is strong due to its simplicity and suitability for scenarios with consistent and non-prioritized disk access.

A user-friendly interface is crucial for the success of a reservation system. Positioning a

Conclusion:

In conclusion, the proposed design and feasibility analysis for the First Come First Served (FCFS) Disk Scheduling algorithm for ticket reservations outline a structured and effective approach to building a reservation system. The combination of intuitive user interfaces, robust backend services, and security measures aims to create a system that not only meets user needs but also ensures the efficient allocation of resources.

The feasibility positioning underscores the practicality of each chosen element within the design, emphasizing the importance of factors such as simplicity, user-friendliness, responsiveness, and security. The FCFS disk scheduling algorithm's role in maintaining a fair and predictable order of reservations aligns well with the goals of the reservation system.

By prioritizing a clean and user-friendly interface, implementing a responsive design, and incorporating key features such as notification services and security measures, the system aims to enhance the overall user experience. Furthermore, the feasibility of integrating logging, reporting tools, and user authentication ensures that the system is not only user-centric but also capable of providing valuable insights and maintaining data integrity.

In the dynamic landscape of ticket reservations, the proposed system is positioned to handle various scenarios effectively, from the initiation of reservation requests to the confirmation and feedback processes. The emphasis on feasibility and careful positioning of elements contributes to the overall success and sustainability of the reservation system.

However, it's important to note that the actual implementation and success responsive adaptation to changing requirements.