USW Musicals Booking System

Module Code: CS4S761 Distributed Computing Faculty of Computing Engineering and Science

Aims and Objectives

The USW Musicals Booking System aims to provide a reliable and efficient platform for managing musical performances and ticket bookings at the University of South Wales. The system implements a microservices architecture to ensure scalability and maintainability.

Key Objectives:

1. Implement a distributed system for managing musical performances
2. Provide real-time seat availability tracking
3. Enable efficient ticket booking for students
4. Ensure system reliability through containerization

System Design

Architecture

The system follows a microservices architecture with three main components: 1. Show Service: Manages musical performances and available seats 2. Booking Service: Handles ticket reservations and booking management 3. Client Application: Provides user interface for students

Key Components

- RESTful APIs for service communication
- Docker containers for deployment
- In-memory data storage for prototype
- Console-based user interface

Technology Stack

- C# (.NET 8.0)
- Docker Desktop
- RESTful APIs
- Docker Compose for orchestration

User Interface

The system provides a console-based interface with the following features: 1. View Available Musicals - Browse upcoming performances - Check seat availability 2. Book Tickets - Select show times - Specify number of seats 3. View Bookings - Track booking status - View booking history

Docker Containerization

Implementation Details

```
# Show Service Dockerfile
FROM mcr.microsoft.com/dotnet/aspnet:8.0 AS base
WORKDIR /app
EXPOSE 80

FROM mcr.microsoft.com/dotnet/sdk:8.0 AS build
# ... build steps ...
```

Docker Compose Configuration

```
version: '3.8'
services:
  showservice:
    build:
      context: ./USWMusicals.ShowService
    ports:
      - "5001:80"
```

Key Commands

```
# Build and run services
docker-compose up --build

# View service logs
docker-compose logs

# Stop services
docker-compose down
```

Challenges and Solutions

Challenges

1. Service Communication
2. Challenge: Ensuring reliable communication between services
3. Solution: Implemented RESTful APIs with proper error handling
4. Data Consistency
5. Challenge: Maintaining seat availability across services
6. Solution: Implemented atomic operations for booking management
7. Docker Configuration
8. Challenge: Proper service orchestration
9. Solution: Used Docker Compose for service management

Future Enhancements

1. Add authentication and authorization
2. Implement real-time notifications
3. Add persistent data storage
4. Develop web-based user interface

References

[To be added using USW Harvard style]