Functional Requirements Implemented

Read Scenario

* Browse theatres currently running the show (movie selected) in the town, including show timing by a chosen date

Write Scenario

* Book movie tickets by selecting a theatre, timing, and preferred seats for the day

Technical Overview:

1)Technologies Used :

a)Framework :Spring Boot(3.2.1),Spring Data JPA

b)Database:H2 ,reason- H2 is a relational database,it is open-source ,and integrates easily with Spring Boot. Very ideal for initial development ,can be easily transitioned to Enterprise Databases like IBM DB2 or Oracle or AWS DynamoDB without modifying the application .

2)CAP Analysis

a)Availability: Since high availability is the desired metric ,rollout to different regions or countries would require the service to scale horizontally.We can provision a container -based deployment for the service and implement a Container Orchestration Service like AWS ECS or use AWS EKS to manage the auto-scaling options. What we can ideally design is having geography-based database sharding and have load balancing mechanisms to distribute requests from different locations to Worker nodes hosting the Movie Ticket Booking service in a Kubernetes cluster, as an example.

b)Performance: We can use caching mechanisms like Redis to store theatre list based on location .This would significantly improve performance .

3)OWASP Top 10

a)Addressing User Authentication: There is a need to provide secure access to the public APIs hosted by the service. This involves User Identity And Access Management.One possible solution could be to have a dedicated Authentication service which provides token-based (eg,JWT) authentication for Users .Using popular protocols like OIDC open connect ,a dedicated Authentication Service can be design ,based on a framework like Spring Authorization Server,which can easily extend the Authorization Code Flow for end users.This service can be deployed on its own or can be integrated in a load-balanced setup ,at the API gateway to help register both the Movie Booking and Authentication service to communicate using a Service Discovery mechanism(either server side like Netflix Zuul or clent side like Eureka).

b)Logging and Monitoring: The distributed nature of the deployment of the service would require a Central Logging and Monitoring solution .Setup of open-source solutions like Opensearch(forked version of ELK) is an option for log analytics as well as monitoring for any defects/downtime in the system .Kibana is a widely preferred option for visualizing the same