**Technologies recommended**

* Language -Java
* Frameworks- Spring boot and hibernate
* Database - PostGres or MySql
* Events processing- Kakfa
* Search engine- Elastic search
* Integration technologies- Docker/kubernets
* Cloud technologies- Amazon or Oracle or some others.
* Preferred editor – Eclipse

**Implemented functional requirements**

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

**Write:** Theatres can create, update, and delete shows for the day.

**Non-functional requirements**

1. Transactional scenarios

* Increase/decrease of movie length effects the show time
* Buying ticket from booking service need to get confirmation from payment service.
* Hold seat in a theater, until booking payment complete. If payment fails or not completed

Release hold seat through Scheduled job which will release the hold seats for other booking.

It can be implemented as scheduled job(with 10 minutes delay) or compensation transaction needs to be executed via API call to release seats.

1. Integrate with theatres having existing IT system and new theatres and localization(movies)

It is directly supported by services which are implemented, we can just add theatre, screen and show details.

1. How will you scale to multiple cities, countries and guarantee platform availability of 99.99%?

* Need to build services as docker images run services behind reverse proxy(like nginx ) with load balancing capabilities in Kubernetes cluster.
* Balancing criteria can be based on country/city/region/number of requests.
* Individual service level scaling.
* Horizontal scalability with more instances running for booking, payment services
* Movie, Theatre service might need less scaling requirements.

1. Integration with payment gateways

* Need to provide multiple payment options like credit card, UPI, Net banking, Wallets, coupons, loyalty points
* Need to store user card or other financial details as per PCI standards.
* Need to have https communication with payment gateways

1. How do you monetize platform?

* Charge percentage fee per each booking, it should be customizable

1. How to protect against OWASP top 10 threats.

* Need to configure reverse proxy to take care on DOD kind of attacks based how requests are coming.
* Need to have token-based authentication/authorization with proper roles defined.
* Once user authenticated need to pass token among services api calls for security.
* Storing or transfer of data like card details, mail id, phone numbers should be encrypted format.
* Need to use AES -256 or more advanced encryption algorithm which provides high security
* Need to use updated software’s which are less vulnerable to knows attacks.
* Write robust APIs which takes care various injections, validate completely before making calls to persistent systems.

**Platform provisioning, sizing & Release requirements**

1. Discuss your technology choices and decisions through key drivers

* I prepared to build this product on micro service architecture to leverage lot of benefits provided by this architecture
* Spring boot as framework of choice to build these services as we are getting lot of built in support in development, testing and lot of flexibility in integrating with various kind of software’s very easily.
* Plan to integrate with elastic search for storing movies and theatres and shows related data into respective elastic search indexes so that search operations can be very quick to the user. Multiple search criteria’s can be implemented easily with elastic search client query apis.
* Use in-memory cache to store master data like locations, cities, movie genre, languages
* API definition and classes can be generated with Open-API generation mechanism.
* Implement observability to monitor health, performance and log aggregation with some kind of ELK or similar choices like fluentD etc.
* Test integration scenarios with mock server like wire mock, to start with.
* Containerize the services with Docker and deploy them in Kubernetes cluster for service orchestration.

1. Discuss database, transactions, and data modelling.

* Services can be deployed on MySQL or postgres DB.
* NoSQL database can be used to maintain reviews, comments, ratings which are non-transactional in nature
* We can use object store mechanism (Oracle object store/Amazon s3) to store movie posters, trailers as they are big size objects.

1. Discuss hosting solution and sizing (Cloud / Hybrid/ Multi cloud)-

Services can be easily deployed in Amazon/Google/Oracle cloud

1. Discuss release management across cities, languages etc

This can be achieved easily across cities with cloud deployments provisioned based on region

1. Provide details on monitoring solution

Enable health, events, logs aggregation with ELK for operations and support.

1. Discuss overall KPIs

* Number of bookings per minute
* Number of failures at payment gateway
* Response time from seat selection to booking confirmation
* Average occupancy per theater/per movie/per show

I have captured all the stories/epics for full working system implementation in a spread sheet.