**Book My Show**

Requirements:

1. Person should be able to book a ticket based on: Current City, Movie, Theatre near him, Select a particular Show, See the seat availability and book
2. System should be able to handle high concurrency
3. System should be secure
4. Other Necessary Features:
   1. Responsive UI (Mobile, WebApp, etc)
   2. Multiple Country/Cities
   3. Payments
   4. Movie Suggestions
   5. Comments & Ratings
   6. Movie Information/Profile
   7. Send tickets by SMS/Email/Whatsapp

---

Theatre should provide us a few APIs which would basically be:

1. Available shows
2. Show Available Seats for a show
3. Book a few seats
4. Lock a seat with a timeout

Person should use BMS and should be able to talk with the DB of the theater which is maintaining the right info for the corresponding theater. There are two ways to connect to the DB (a) Directly talk with the DB, (b) Through the APIs provided by the movie theater.

I think (b) is better since we can talk with the server and call the relevant API to book the ticket.

A picture containing text, whiteboard

Description automatically generated

Diagram

Description automatically generated

**System Design:**

1. Mobile App/Web App would talk with the load balancer which has different techniques to balance the load between App Servers (Consistent Hashing, Round Robin, Weighted Round Robin).BMS uses NGINX Load Balancer.
2. CDN: used to cache video, images, or APIs.
3. App Servers: This is used to call the APIs. BMS uses Spring Boot, Swagger and Hibernate.
4. ELK: Searching a movie/shows, we can dump all the data into Elastic Search. It is restful search APIs in the system.
5. Cache: To save all the info related to the movie, seat ordering, place, etc. Redis (Distributed)
6. RDBMS: Country -> City -> Theatre -> Screens -> Seats, Movie, Handle Transactions. This can be used in Master – Slaves Architecture where Slaves can b sued for read while Master can be used for Write. BMS uses Mysql.
7. NOSQL: Movie Information, Comments, Reviews. This can be distributed/sharded by Geo. We can set the replication factor and consistency level.
8. Async Workers: Send email, sms, whatsapp are all 3rd party API calls which take a lot of time and can not be done sync and must be done ASYNCRONOUSLY. The app server once it confirms a ticket will put the message in a queue and then the workers would pick it up from the queue and then process the task. BMS uses RabbitMQ.
9. GCM Notifications can also be used to send notifications on to the user’s movie and desktop app. Python Celery can be used.
10. Recommendation Engine: Hadoop Platform to get the B.I. All the logs, user activities should be dump into Hadoop on top which we can run HIVE, PIG queries to extract some information like user behavior for recommending movies.
11. Spark: We can dump the data into Spark/Storm to do some real time analysis (Trends right now).
12. Payment Gateway: 3rd party Payment Gateway can be integrated like JUSPAY which can provide Wallets, Netbanking, Debit, Credit Card payments.
13. BMS is totally on AWS. It has microservices architecture.

**What happens when a customer tries to Book a ticket?**

1. Customer uses the Location filter via GPS tracker of phone and then clicks on a Movie, then clicks on Shows and selects a show and seats.
2. BMS makes a call to the theater DB to lock the seats for 10 mins.
3. Customer makes the payment and gets the callback from the Payment Service and we notify the user regarding the booking.
4. A ticket is generated and sent to the user with the details, QR code, etc.

**Database:**

SQL DB:

Diagram

Description automatically generated

NOSQL DB:

Diagram

Description automatically generated