Problem statement

XYZ wants to build an online movie ticket booking platform that caters to both B2B (theatre partners) and B2C (end customers) clients.

Key goals it wants accomplished as part of its solution:

- Enable theatre partners to onboard their theatres over this platform and get access to a bigger customer base while going digital.

- Enable end customers to browse the platform to get access to movies across different cities, languages, and genres, as well as book tickets in advance with a seamless experience.

Technologies recommended

- Language -Java and other add-on languages
- Frameworks- Any
- Database - Any
- Integration technologies- Any
- Cloud technologies- Any
- Preferred editor to build and present solution

PS: ***The given exercise will help you get prepared for technical discussion and demonstrate your current understanding on key architectural artifacts. It's expected that you may not all sections hence can ignore them.***

Evaluation criteria

- Code artifacts (**APIs Contract, Design Patterns,** Any one Scenario Implementation)
- Design principles to address functional requirement and non-functional requirement
- DB & Data model
- Platform solutions detailing
- Solution completeness, presentation, and discussion.
- Solution coverage uniqueness and extensibility.

Note: Incomplete solution component would be discussed during discussion round. All sections are not mandatory.
You can skip solution areas that you are not comfortable by making a note of it.

Functional features to implement Good to have - Code Implementation (Read scenario)):

Anyone of the following read scenarios: (Only Service Implementation needed/ No UI required)

- Browse theatres currently running the show (movie selected) in the town, including show timing by a chosen date
- Booking platform offers in selected cities and theatres
    - 50% discount on the third ticket
    - Tickets booked for the afternoon show get a 20% discount

Candidate Solution:

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

- Search for running & upcoming movies for specific location,	Search	City * Movie Name * Genre or Language	- Perform efficient search using Elasticsearch index (shows data)

genre in specific movie theatre		Booking Date (Default Current Date) *	

Panel Feedback:

Anyone of the following write scenarios: Good to have - Code Implementation (write scenario):

- Book movie tickets by selecting a theatre, timing, and preferred seats for the day
- Theatres can create, update, and delete shows for the day.
- Bulk booking and cancellation
- Theatres can allocate seat inventory and update them for the show

Candidate Solution:

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

Book Movie Ticket	Ticket	Location (City)* Movie Name * Theatre Name * Theatre Hall * Show Timing (s) * - Seating Plan - Section Name * (Currency as City/ Country) - VIP, Executive & General -- Price - Block Seats [For offline purchase using section specific Seating Plan]	- Customer select specific show from specific theatre in particular city - Customer also able to see available & already booked seats for that particular show along with section specific prices - Customer can select number of tickets (up to max limit in one booking) & select those seats from a specific section (VIP or Executive or General) - Customer can confirm the booking & proceed towards the payment processing - Once Customer clicks on the Confirm Booking, those selected seats from a specific section will be blocked for 10 mins (so no other person can book the same seats in that duration). If a customer successfully completes the payment within 10 minutes, those seats marked as booked one & no longer available else if customer fails to complete the payment within 10 minutes then those blocked tickets will be available again for customers. - Email & SMS notifications will be sent out to customer with all the details of booked seats, show timings of that movie & QR code to be scanned at the movie hall for your check-in.

Panel Feedback:

Discussion topics & Logical View:

Non-functional requirements-(Mandatory -Design/Arch solution & Optional Implementation**):**

- Describe transactional scenarios and design decisions to address the same.
- Integrate with theatres having existing IT system and new theatres and localization(movies)
- How will you scale to multiple cities, countries and guarantee platform availability of 99.99%?
- Integration with payment gateways
- How do you monetize platform?
- How to protect against OWASP top 10 threats.

Candidate Solution:

- Describe transactional scenarios and design decisions to address the same.
    - **Same Seat selection from different customers at same time**:
        - Once Customer clicks on the Confirm Booking, selected seats from a specific section will be blocked for 10 mins (so no other person can book the same seats in that duration).
        - If a customer successfully completes the payment within 10 minutes, those seats marked as booked one & no longer available else if customer fails to complete the payment within 10 minutes then those blocked tickets will be available again for customer.
        - We can either use theatre databases or API's where this sort of locking/ unlocking mechanism is already in place or we can build our locking/ unlocking system (optimistic locking) to cater those theatres which don't have any sort of online presence.
- Integrate with theatres having existing IT system and new theatres and localization(movies)
    - To integrate with any of theatres having their existing system, we need to get align on couple of items:
        - Get access to their system by registering our system as a new client into their system & acquire a set of api key, access & secret keys. Use AWS KMS service for keeping all such keys more secure & assign specific actions on specific services.
        - Get database access with restricted DML access (DDL not required)
        - Align on seat reservation template & strategies so they handle offline seats accordingly
        - Align on shows data import or sync into our system using scheduled jobs or providing manual access of our system to them
        - To cater localization, we need to identify what sort of data will support localization & plan to keep it segregated accordingly. Mostly movie related metadata, we need to handle specifically as images & video trailers will not have any differentiation.
- How will you scale to multiple cities, countries and guarantee platform availability of 99.99%?
    - For high availability, we can keep multiple instances of multiple services in 2-3 availability zones & use ELB (Elastic Load Balancer) to distribute load into them accordingly. In case of increase or decrease, AutoScaling groups can scale up or scale down servers respectively.
    - We can use database sharding on the basis of country specific data segregation rule, create ElasticSearch configure for country specific data & deploy those servers in nearest data centers.
- Integration with payment gateways
    - Payment Gateway integration in any application are mostly follow same approach
        - Register your application in specific payment gateway provider which supports debit card, credit card, upi, wallet (provider specific if any supports like PayTM)
        - Get you client id, api keys & set your callback url (which will be used for marking any transaction as success or failure once payment processed at bank)

        - Using specific keys & order details (must require 1 unique identifier which helps us to track status at a later stage along with payment details) we can initiate transactions and get transaction id.
        - Using that transaction id we can check transaction status at a later stage & also that will help in Refund Transaction as well for absolute tracking.
        - Always use **Strategy Design pattern**, to handle and support multiple payment providers
- How do you monetize platform?
    - Integrating with advertisement providers & selling out your premium space (homepage banners slots) or white space (sidebar or footer)
    - Providing an option to sell out offers, vouchers & gift cards (myntra, amazon, etc) & get commission from the respective businesses accordingly.
    - Promoting local sellers or business outlets as per city selection or as per theatre selection page.
- How to protect against OWASP top 10 threats.
    - Implementing robust RBAC rules on infrastructure resources & application resources
    - Providing security on data in rest (encrypt sensitive information) & transit (use of HTTPs)
    - Provide robust validation layer before any sort of data reaches our business layer & prevent prepared statements instead of raw SQL queries
    - Check for vulnerabilities in the code or dependency using tools sonarqube, nexus, etc.
    - Setup rules for API rate limiting, throttling, security using any API gateway.
    - Setup robust Authentication & Authorization layer to prevent any unwanted access.
    - Keep our servers (managed by us) updated to avoid any vulnerabilities using bastion hosts only.
    - Logging & Monitoring of the application plays a crucial role & setup logging process with all essential details to track down any form of critical failure (CloudWatch & CloudTrail).

Panel Feedback:

Discussion topics & Logical View:
Platform provisioning, sizing & Release requirements: (Mandatory-Architecture artifacts)

- Discuss your technology choices and decisions through key drivers
- Discuss database, transactions, and data modelling.
- Discuss enterprise systems that you may need to manage specific areas.
- Discuss hosting solution and sizing (Cloud / Hybrid/ Multi cloud)- Any
- Discuss release management across cities, languages etc
- Provide details on monitoring solution
- Discuss overall KPIs
- Create a high-level project plan and estimate breakup.

Candidate Solution:

- Discuss your technology choices and decisions through key drivers
    - **Spring Boot (Java 8+)** - RAD tool for developers & keeps our services lightweight & deployable
    - **MySQL (RDBMS)** - To maintain the relationship & data flow among entities like TheartrePartner, Theare, TheatreHall, Movies, Language, Genre, Locations, Shows, Bookings, etc keeping ACID constraints in place

    - **Elasticsearch (NoSQL) -** Effective & fast solution for searching shows for particular city or other filter criterias. Also a very efficient solution to store non structured data like movie metadata for different languages, user specific movie ratings & reviews, log the user specific activities for helping out to build recommendation engine.
    - **Apache Kafka**: Effective solution to publish & consume event specific notifications (like sending push notification, email & SMS) in asynchronous manner to improve overall application performance. Highly scalable & fault tolerant in nature if configured properly.
- Discuss database, transactions, and data modelling.
    - Refer to the provided **ERD.**
- Discuss enterprise systems that you may need to manage specific areas.
    - ***TBD***
- Discuss hosting solution and sizing (Cloud / Hybrid/ Multi cloud)- Any
    - I have used couple of different AWS API & solutions in multiple applications
        - **EC2**: We can evaluate our customer load for the initial 3 months of the product launch using load testing & pick up the suitable EC2 instances (vary as per CPU, RAM, Hard Disk).
        - **S3**: This is quite a cheap service with high availability. We can use its standard type & suitable for all sorts of images (movie, cast, crew) & videos (trailers). Once any image or video gets uploaded, we can trigger a lambda function to create copies of the same image in multi resolution & optimized formats.
        - **RDS:** We can use master/ slave strategy here to segregate write & read operations of different machines & setup sync process between master and slave to get eventual consistency. As it is a costly service, we should keep data archival policy ready & keep our database lighter after a certain period of time. Archived data can be moved to S3 glacier if we require it anytime in future.
        - **Elasticsearch:** Effective & highly scalable NoSql solution. We can keep multiple indexes for per country. We can use a good archival strategy (same as mentioned in **RDS**) to keep our indexes lighter.
        - **CloudFront**: It's always better to use cache based solutions for providing static & media content to customers from their nearest edge location. It also increases a little cost so we can plan to remove the content or expire the cache (set TTL) once the actual content is archived.

- Discuss release management across cities, languages etc
    - ***TBD***
- Provide details on monitoring solution
    - We can use **CloudWatch** & **CloudTrail** services for logging all activities & we can monitor any unwanted activity happening in any of the servers.
    - We can set specific alerts & notifications in case of known or unknown incidents. Like we can set alerts & notification if CPU usage goes beyond 75% in the last 30 minutes, etc.
    - We can also trigger or automate some process like run a lambda to run 1 more EC2 server or run another specific service.
- Discuss overall KPIs
    - Conversion rate (how many people are visiting site & how many actually booking the ticket)
    - Page specific visits (which page is mostly visited for how long & which page is least visited)
    - Customer feedback about platform
    - Track offers attracting or helping in conversion rate
    - Track of mostly used payment mode

    - Theatre boarding process (conversion rate of theatres) & coordination
    - Theatre Partner feedback about platform
- Create a high-level project plan and estimate breakup.
    - Refer to the attached **Excel sheet (Publicis Sapient Excercise)**

Panel Feedback:

Discussion topics & Logical View:
Product management and Stakeholder management

- Please talk about stakeholder management instances
    - What decisions and actions were taken for decision closure?
- Overall technology management
- Enabling team and introducing efficiencies
- Delivery planning and estimates

Candidate Solution:

- Please talk about stakeholder management instances
    - What decisions and actions were taken for decision closure?
        - Keep everything transparent & to the precise point of requirement understanding
        - Discuss every requirement using appropriate wireframes, UI mockups & workflow diagrams with stakeholders and lock the document by taking sign off from them
        - Re–visit or revise any requirement which is recently reviewed & not signed off due to change in the business ask or ambiguity in understanding.
        - Make sure to set the priority of key items to be delivered (sprint wise) & get an alignment on the same with stakeholders before committing everything in one go
        - In terms of resources (team, tools & infrastructure) for releasing specific features in specific environments (dev, qa, uat, prod), there should be detailed break down shared with all stakeholders
- Overall technology management
    - Prepare all require design diagram before starting any sort of diagrams (System Architecture, Use Case, Sequence, ERD, etc) and shared with all the team
    - Prepared checklist of tools, resources, processes to be used by every team member working on same project (Docker can help us here to setup same DEV env)
    - Peer review & team code review should be in place to improve code quality & maintainability
    - Follow the TDD approach at least unit & integrated test
- Enabling team and introducing efficiencies
    - Keep team motivated by praising the good work or any new initiatives (in front of team)
    - Always be transparent (as much you can) & set the expectations clearly with them
    - Feedback should be given/ taken in constructive manner (one to one talk)
    - Try to learn something new and share or emphasize team to do the same while working on any new task or user story
    - Knowledge sharing is always enhance the skills of team & strengthen the team bonding
- Delivery planning and estimates
    - List out priority items from product backlog after confirmation with key stakeholder
    - Discuss those items with team & ask team to prepare sprint backlog (items going to be released)
    - Team will decide on the estimates as per the available resources & tools

- We can review the provided estimates with team & adjust (increase/ decrease) estimates accordingly (realistic adjustment)

Panel Feedback: