**Problem statement**

XYZ is a start-up, an online movie ticket booking platform, providing with B2B and B2C solution for its theatre partners and end customers to onboard theatre and book online tickets.

B2B Theatre partners can onboard their theatres over this platform to go online and get access to bigger customer base and become online.

B2C end customers can browse platform to get access to movies across different cities, languages and genre to be able book ticket in advance and have a seamless experience.

**Technologies recommended**

* Language -Java and other add on languages
* Frameworks- Any
* Database - Any
* Integration technologies- Any
* Cloud technologies- Any
* Use your own editor to build and present solution

**Evaluation criteria**

* Code Implementation and completeness (APIs and Design Patterns)
* Design Principles to address functional requirement and Non-functional requirement
* Platform Solutions detailing
* Solution completeness, presentation and discussion.
* Solution coverage uniqueness and extensibility.

Note: Incomplete solution component would be discussed during discussion round.

*You can skip solution areas that you are not comfortable by making a note of it.*

**Functional features to implement** (Mandatory - Code Implementation):

**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

**Anyone of the following write scenarios:** (Only Service implementation needed-No UI required)

* 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 & Cancel booking
* Theatres can allocate seat inventory and update them for the show

**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.

**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 KPIs overall.
* Create a high-level project plan and estimates breakup.

Disclaimer:

*This document is meant to assess your technical skills and is classified as "Sapient confidential". This document by any means shall not be used/shared without permission from Sapient, non-adherence to this can get your candidature blocked for employment with Sapient.*

**TOC**

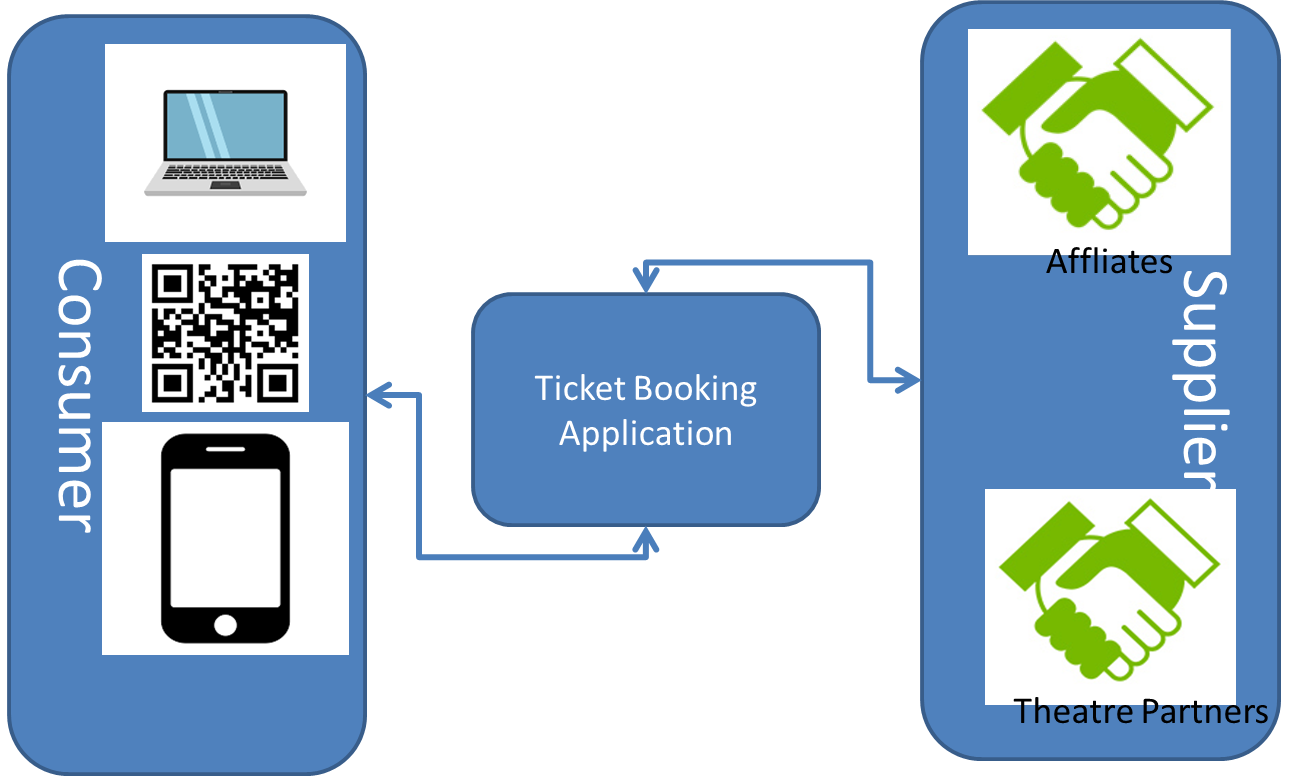
* High Level Design of BookMyTicket Application
* Technology Stack
* Deployment Model
* Application Monitoring
* Non Functional Requirements
  + Performance
  + Web Application Security
  + Load Balancer for Horizontal Scaling
  + MicroServices architechture for High Availability
* Functional Requirements
  + Partners Onboarding Service
  + Customer Onboarding Service
  + Ticket Booking Service
  + QRScanner Service
  + Notification Service
  + Asynchronous Partner Onboarding (covers integrating partner infrastructure)
* Low Level Design for Functional Requirements
* CI/CD Pipeline (Git/BitBucket/Unit Test/Maven/SymLink)
* OWASP top 10 threats
* Monetize
  + Ads
  + Affliate Program
* High Level Project Plan
* KPI

Please note that there are many other layers to think about given the time trying best to cover the most usual cases thereby may be missing finer details, which can be up for discussions as well.

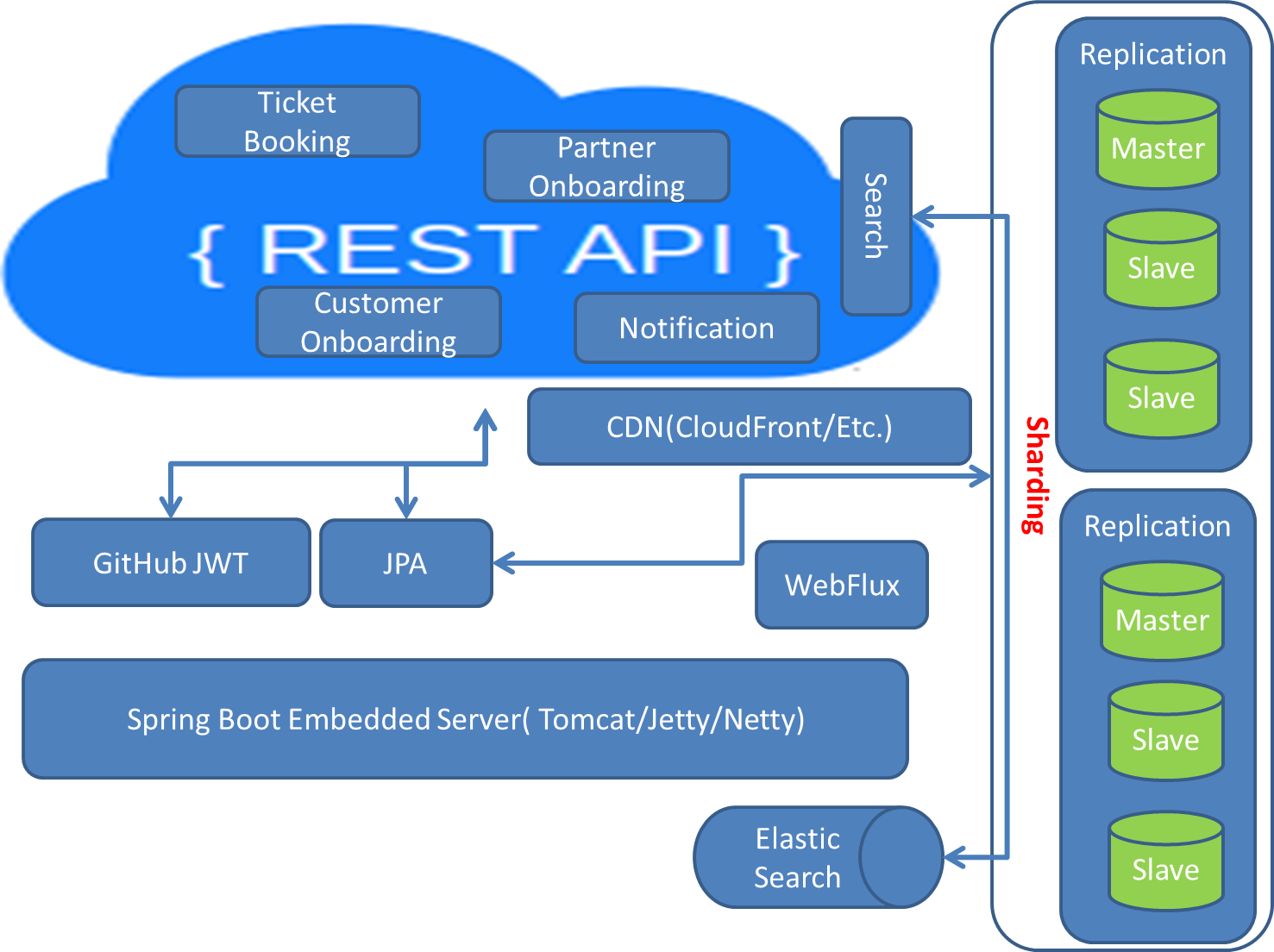
**HIGH LEVEL DESIGN OF THE BOOKING SYSTEM**

There are 3 types of consumers in this application. They are:

* Desktop/Laptop User – Generally searching the movies or else buying a ticket.
* Mobile User – Same as Desktop/laptop user.
* Data Providers
  + Theatre Partners
  + Affiliates – Review writers, blog writers etc.

****

**TECHNOLOGY STACK**

****

**GitHub JWT**

To enable security both at the user access (Partners as well as Consumers) one can use any OAuth2 provider service like Gihub or Google OAuth2 for web security. The other option as well is to have own implementation of resource server using Spring Boot Authorization Server.

**WebFlux**

This one we can use for async notification to customers about offers upcoming bookings etc. With the use of reactive web we can achieve that. Note reactive web is based on reactor framework.

**CDN**

This will be used for images for high performance.

**MySQL Cluster**

While all the booking data, partner data, payment data and customer data needs transactional behaviour and hence RDBMS is a good choice for storing them. It a good idea to create replica mysql cluster for high availability of data just in case one of the master or slave goes down. Imaging the consumer is about to get to the screen the watch the movie and the database is down the verify the QRCode provided by the Booking Service, this will be huge customer dissatisfaction which can be avoided using replication. If the data set grows very large one can go over data sharding where data is distributed across the sharded clusters.

**Elastic Search**

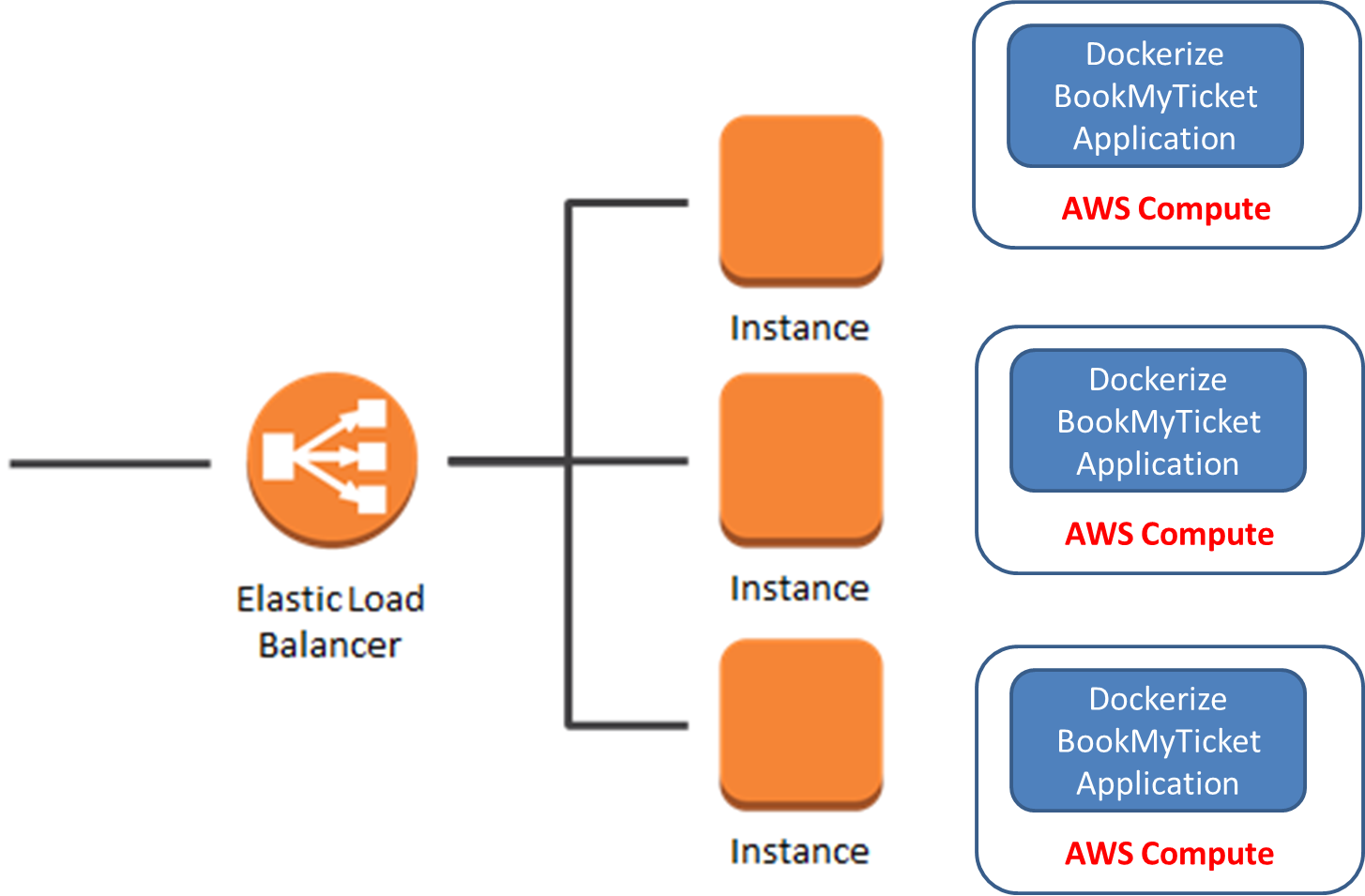
This will be used to store all movies data, reviews etc. User search will be fetched from Elastic search.

**Payment Gateway**

This can be achieved via registering your application with Payment Gateway Providers and when the user is prompted for payment, the user is taken to the gateway provider website for payment and on success redirected back to the main application for confirmation/rejection etc.

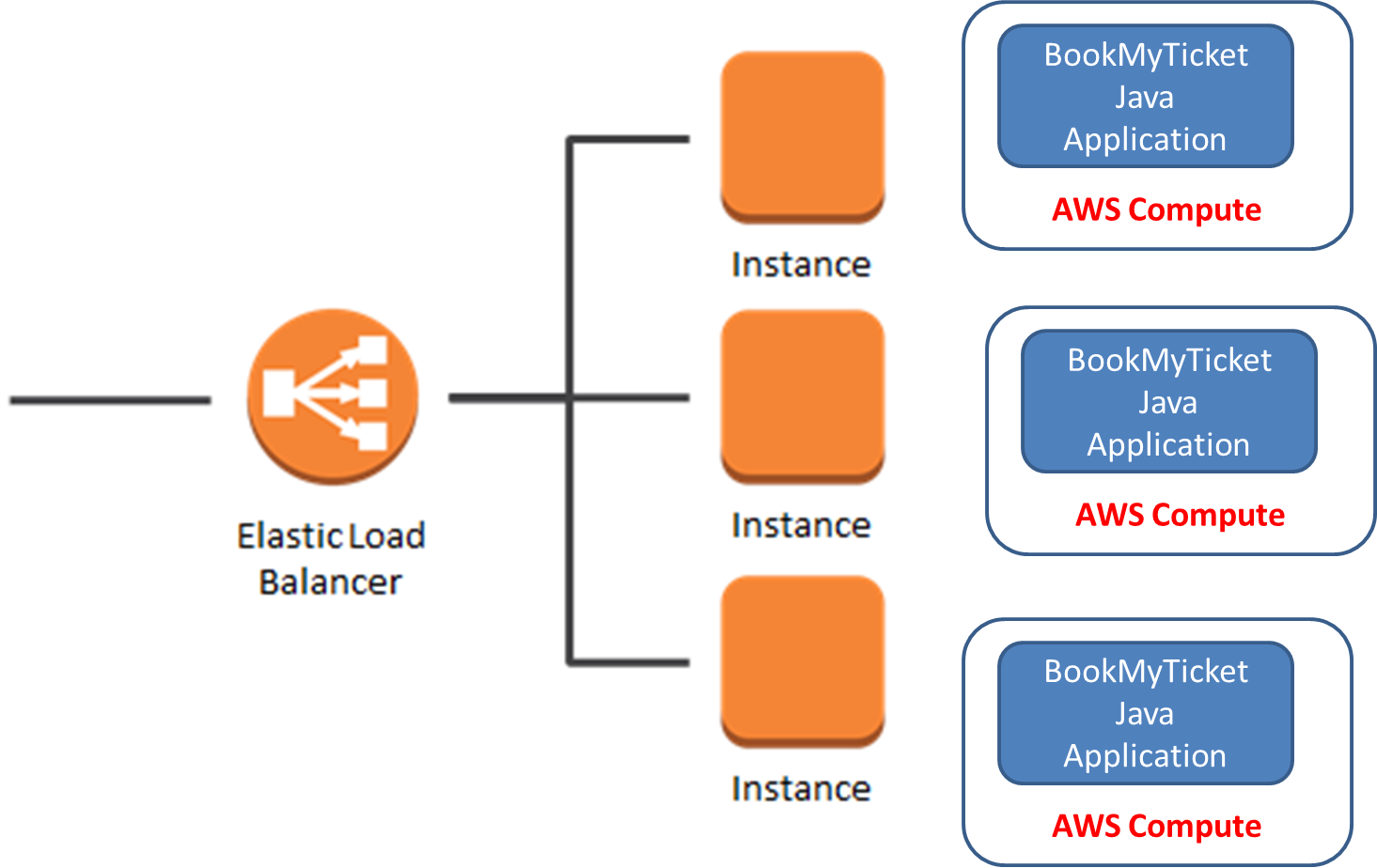
**Application Deployment Model**

**Option 1 – Dockerize and Deploy**

****

**If you dockerize the application, we can get all docker advantage.**

**Option 2 – Java Application Deployment (Why Not Dockerize – Knowledge/Resource/DevOps)**

****

**Other Deployment Options**

* **In house deployment (Cons – Infrastructure Management, Pros – Cost Advantage)**
* **Hybrid Model**

**Application Monitoring**

**TODO**

**Non Functional Requirements**

**TODO**

**Functional Requirements**

**TODO**

**Low Level Design for Functional Requirements**

**TODO**

**CI/CD Pipeline (Git/BitBucket/Unit Test/Maven/SymLink)**

**TO BE DISCUSSED**

**OWASP top 10 threats**

**TODO**

**Monetize**

**TODO**

**High Level Project Plan**

**TODO**

**KPI**

**TODO**