Oyo Hotel Bookings

# Problem statement

Oyo Hotels helps customers to surf different hotels available and allows them to book hotel rooms in different cities. The application will show hotel details and also the customer/user can choose the hotel and book the hotels.

The following section will cover aspects related to Oyo Hotel Bookings

* View All hotel details
* Make a booking
* View All the booking details

**Scope of the System**

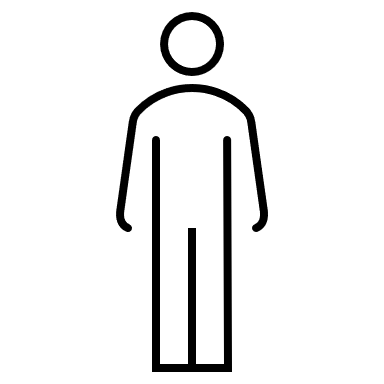
The scope of the system is explained through its modules as follows

View All hotel details – The system will show the details of available hotels in different cities.

Make a booking – The system can help the user with booking the hotels from the available list.

View All Hotel bookings -- The app shows what are the hotel booked on what dates.

**Use Case Diagram**



**Flow Diagram**

# Project Development Guidelines

The project to be developed based on the below design considerations

|  |  |
| --- | --- |
| **Backend Development** | * Use Rest APIs (Springboot/ASP.Net Core WebAPI to develop the services * Use Java/C# latest features * Use ORM with database * Use Swagger to invoke APIs * Use JWT for authentication in SpringBoot/WebApi. A Token must be generated using JWT. Tokens must expire after a definite time interval, and authorization must be handled accordingly based on token expiry * Implement Logging * Implement API Versioning * Implement security to allow/disallow CRUD operations * Message input/output format should be in JSON (Read the values from the property/input files, wherever applicable). Input/output format can be designed as per the discretion of the participant. * Any error message or exception should be logged and should be user-readable (not technical) * Database connections and web service URLs should be configurable * Implement Unit Test Project for testing the API * Follow Coding Standards |
| **Frontend Development** | * Use Angular/React to develop the UI * Implement Forms, databinding, validations * Implement Routing and navigations * Use JavaScript to enhance functionalities * Implement External and Custom JavaScript files * Implement Typescript for Functions, Operators. * Any error message or exception should be logged and should be user-readable (and not technical) * Follow coding standards * Follow Standard project structure |

# Good to have implementation features

* Generate a SonarQube report and fix the required vulnerability
* Use the Moq framework as applicable
* Create a Docker image for the frontend and backend of the application
* Implement design patterns
* Deploy the docker image in AWS EC2 or Azure VM
* Build the application using the AWS/Azure CI/CD pipeline. Trigger a CI/CD pipeline when code is checked-in to GIT. The check-in process should trigger unit tests with mocked dependencies
* Use AWS RDS or Azure SQL DB to store the data