IIHT

FLIGHT BOOKING

Contents

[1 Problem Statement 2](#_Toc70929966)

[2 WIREFRAMES 3](#_Toc70929967)

[3 Application Architecture 4](#_Toc70929968)

[3.1 Microservice Architecture (Compute and Integration/Presentation/Networking and Content Delivery): 4](#_Toc70929969)

[4 Cloud Architecture 5](#_Toc70929970)

[5 Tool Chain 5](#_Toc70929971)

[6 Business Requirements: 7](#_Toc70929972)

[7 Proposed Rest Endpoints to be exposed 8](#_Toc70929973)

[7.1 Rest APIs: 8](#_Toc70929974)

[8 Key Rubrics/Expected Deliverables 8](#_Toc70929975)

[8.1 Debugging & Troubleshooting 9](#_Toc70929976)

[8.2 Code Quality/Optimizations 9](#_Toc70929977)

[9 Platform 9](#_Toc70929978)

[10 Methodology 9](#_Toc70929979)

[10.1 Agile 9](#_Toc70929980)

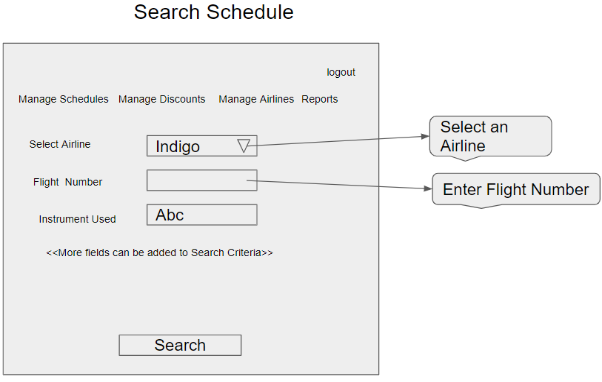
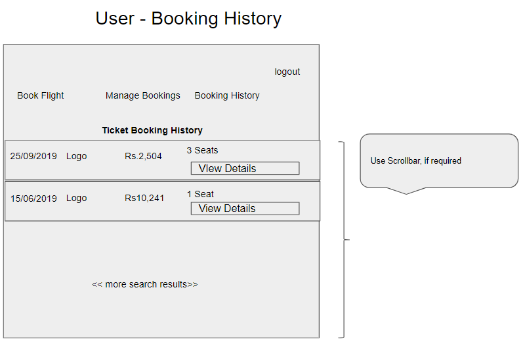
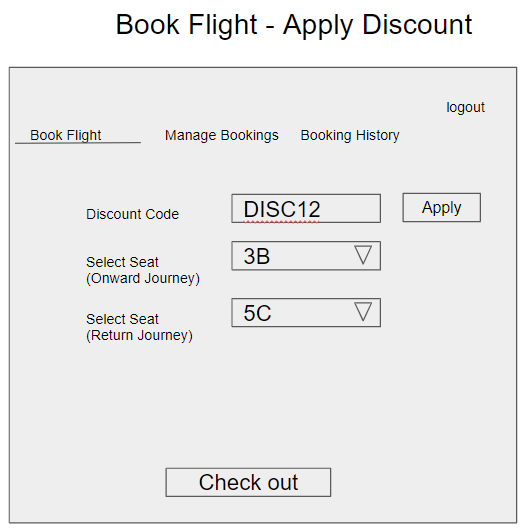
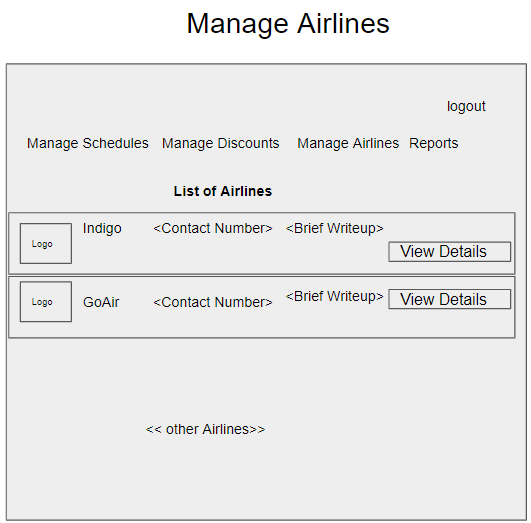
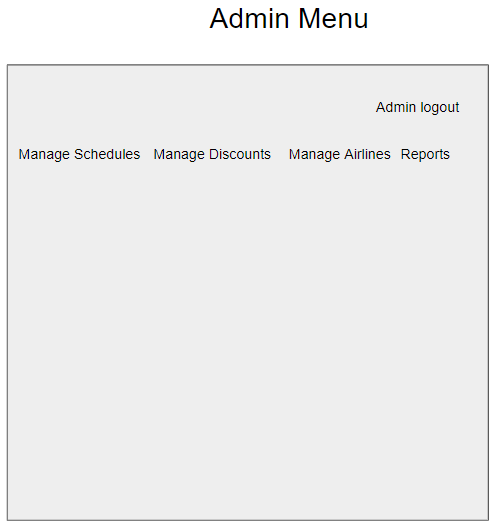
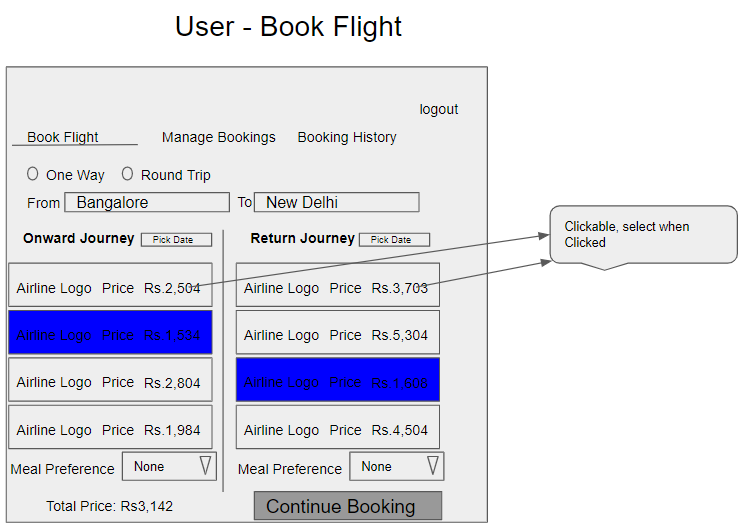
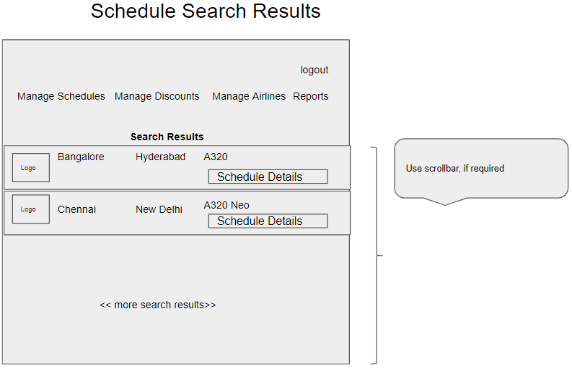
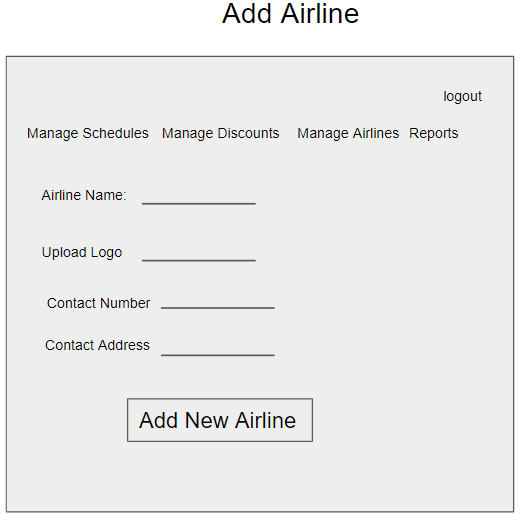
# Problem Statement

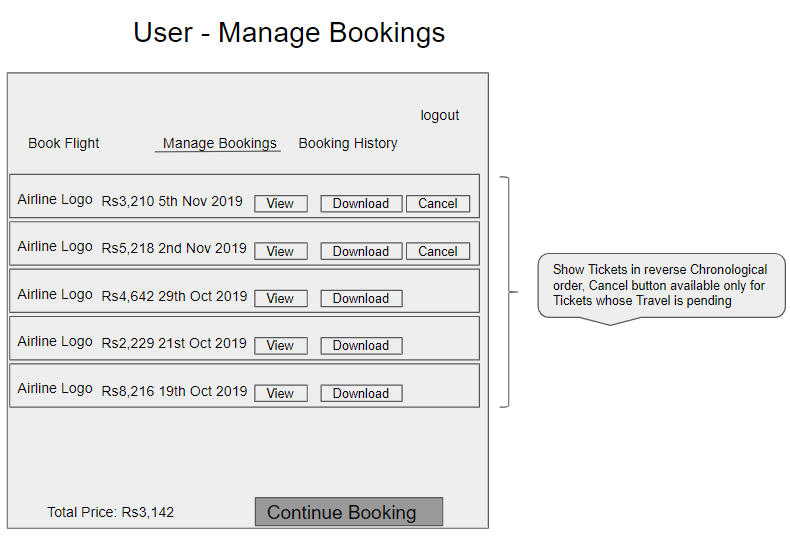
Build a software system which lets user search for a Flight Ticket and book it & includes Admin related activities. User can also cancel or update the Ticket. Below are the different roles, which need to be supported by above Software System.

1. User
2. Admin

The scope includes developing the application using tool chain mentioned below.

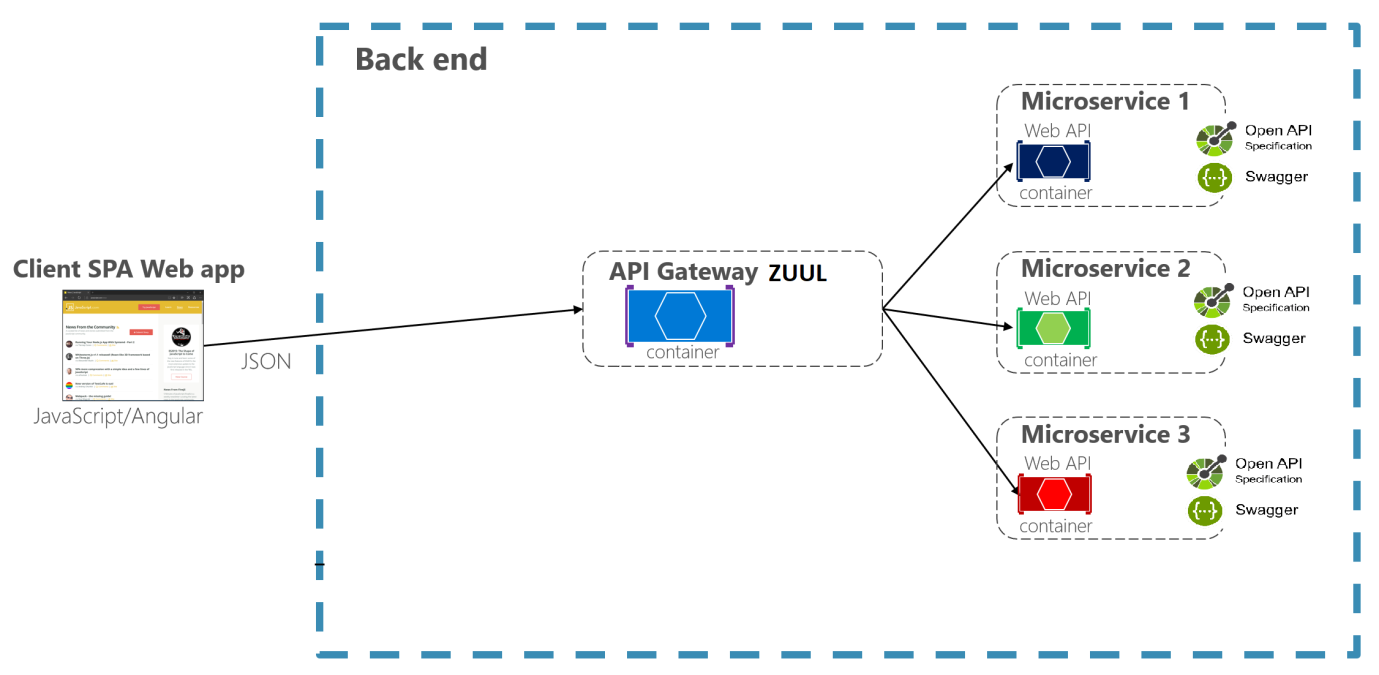
# WIREFRAMES

UI needs improvisation and modification as per given use case.

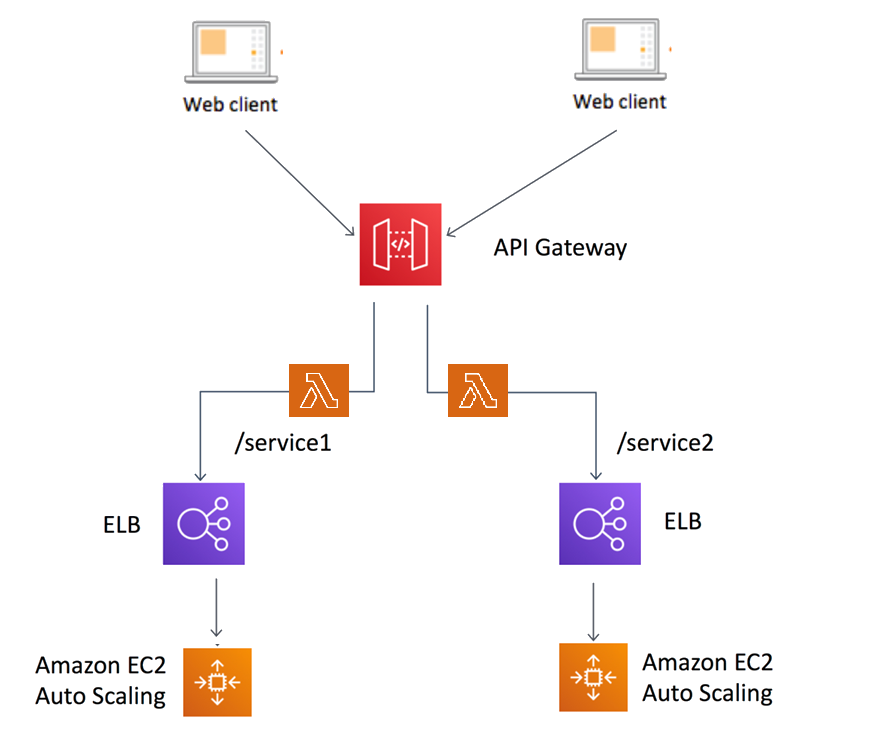


# Application Architecture

## Microservice Architecture (Compute and Integration/Presentation/Networking and Content Delivery):



# Cloud Architecture



# Tool Chain

|  |  |  |
| --- | --- | --- |
| Competency | Skill | Skill Detail |
| Engineering Mindset | Networking and Content Delivery |  |
|  | DevOps |  |
|  | Secure Coding | Veracode |
|  | Code Quality | Sonar |
| Programming Languages | Application Language | Java |
| Products & Frameworks | Presentation | Angular |
|  |  | Material/Bootstrap, rxjs, ngrx/store |
|  |  | Javascript/Typescript |
|  | Networking and Content Delivery | Zuul |
|  | Security and Identity | OpenIAM |
|  | Compute & Integration | Spring Boot |
|  |  | Kafka |
|  | Database & Storage | MySQL |
|  | Governance & Tooling | Git |
|  |  | Junit |
|  |  | Mockito |
| Engineering Quality |  |  |
| Platform | Cloud Tools | AWS EC2 |
|  |  | RDS-MySQL/Aurora |
|  |  | AWS Lambda |
|  |  | AWS API Gateway |
|  |  | AWS ELB(Elastic Load Balancer) |
|  |  | AWS CloudWatch |

# Business Requirements:

As an application developer, develop microservices with below guidelines:

|  |  |  |
| --- | --- | --- |
| User  Story # | User Story Name | User Story |
| US\_01 | User Mode | 1. User can search for a Flight based on date/time, from place/to place, one way or round trip 2. Each Search result need to display Flight Date/time, Airline Name/Logo, Price(to & round trip – TBD) 3. From Search results, User should be able to select a specific Flight and go ahead and complete Ticket Booking by providing below details  * Name and Email ID * Number of seats to book. * Details of each passenger (NAME:GENDER:AGE) * Opt for Meal(Veg/Non veg) * Select Seat Number(s)  1. On successful Ticket Booking, PNR number need to be generated, it should be possible to download TicketBooking can be done a Logged in User only 2. With email id user should be able to  * view History of Ticket Bookings, * Cancel a Ticket only prior to a day(24 hrs) before journey date.  1. With PNR number view the booked ticket details |
| US\_02 | Admin Mode | 1. Admin shall be able to login/logout. 2. There can be pre-defined username/password for Admin. 3. Admin shall be able to add/block an Airline. When Airline is Blocked, Flights belonging to that Airline will not be shown in Ticket Search results. 4. Admin shall be able to add Inventory/Schedule of an existing Airline by specifying below details:  * flight number * Airline * From Place * To Place * Start date time, * End date time, * Scheduled Days(Daily, Week Days, Week Ends, For specific days specify the list of Days like Mon, Wed) * Instrument used(A320, A320 neo, etc…) * Total number of business class Seats * Total number of non-business class Seats * Ticket cost (consider taxes and other charges), * number of rows, * meal(none, veg, non veg) |

# Proposed Rest Endpoints to be exposed

## Rest APIs:

|  |  |  |
| --- | --- | --- |
| **POST** | **/api/v1.0/flight/airline/register** | **New airline booking** |
| **POST** | **/api/v1.0/flight/admin/login** | **Admin login** |
| **POST** | **/api/v1.0/flight/airline/inventory/add** | **Add Inventory/Schedule of an existing Airline** |
| **POST** | **/api/v1.0/flight/search** | **Searches for flight** |
| **POST** | **/api/v1.0/flight/booking/{flightid}** | **Book ticket** |
| **GET** | **/api/v1.0/flight/ticket/{pnr}** | **Get Booked ticket details based on PNR** |
| **GET** | **/api/v1.0/flight/booking/history/{emailId}** | **Get Booked tickets history based on Email ID** |
| **DELETE** | **/api/v1.0/flight/booking/cancel/{pnr}** | **Cancel a booked ticket** |

# Key Rubrics/Expected Deliverables

As an application developer:

* 1. Develop the application as a microservice architecture.
  2. Ensure package Structure for project is like com.flightapp.\* with proper naming conventions for package and beans.
  3. Use application.properties or yaml file to maintain all spring boot config.
  4. Implemented the package structure - Controller, Interface, Service, DAO, Testing, Validation, Security etc
  5. Implementation as follows:
     1. Use Domain Driven Design to implement distributed architecture
     2. Follow Single Data Store per microservice practice
     3. Document REST endpoints with OpenAPI/ Swagger
     4. Add CQRS pattern for Event Sourcing using Kafka for saving and retrieving flight details, using Kafka (producer & consumer) topics
     5. Expose all rest Endpoints using a common API Gateway Zuul
  6. Secure all Rest EndPoints by configuring SSL Certificate for Cloud
  7. Use OpenIAM to add OTP based two factor authentication for secured operations

## Debugging & Troubleshooting

1. Generate bug report & error logs - Report must be linked with final deliverables which should also suggest the resolution for the encountered bugs and errors.

## Code Quality/Optimizations

1. Associates should have written clean code that is readable
2. Associate should have used the Code Analyzer (PMD/SonarQube) to ensure code quality and standard code style.

# Platform

Use User Story-1 from the Business Requirements to implement the below.

* + - 1. Use EC2 to deploy application on cloud.
      2. Use RDS-MySQL / Aurora(SQL) as a database for the Application.
      3. Use AWS Lambda and DB to build a backend process for handling requests for Flight booking App.
      4. Use Amazon API Gateway to expose the Lambda functions built in the previous step to be accessible on public internet.
      5. Use AWS ELB(Network Load Balancer) to configure the load balancing of the instances

Note : Minimum 2APIs (UI+Backend) to be hosted in cloud

# Methodology

## Agile

1. As an application developer, use project management tool along to update progress as you start implementing solution.
2. As an application developer specify the estimation and planning as a part of Agile process.
3. As an application developer, the scope of discussion with mentor is limited to:
   1. Q/A
   2. New Ideas, New feature implementations and estimation.
   3. Any development related challenges
   4. Skill Gaps
   5. Any other pointers key to UI/UX and Middleware Development