**Flights Booking Management System Mini Project**

**Contents**

[**1 Problem Statement 2**](#_heading=h.30j0zll)

[**2 WIREFRAMES 3**](#_heading=h.1fob9te)

[**3 Application Architecture 4**](#_heading=h.3znysh7)

[**3.1 Microservice Architecture (Compute and Integration/Presentation/Networking and Content Delivery):**](#_heading=h.2et92p0)

[**5 Tool Chain 5**](#_heading=h.3dy6vkm)

[**6 Business Requirements: 7**](#_heading=h.1t3h5sf)

[**7 Proposed Rest Endpoints to be exposed 8**](#_heading=h.4d34og8)

[**7.1 Rest APIs: 8**](#_heading=h.2s8eyo1)

[**8 Expected Deliverables**](#_heading=h.17dp8vu)

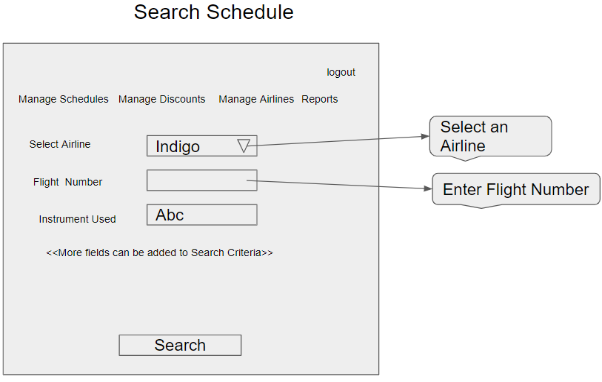
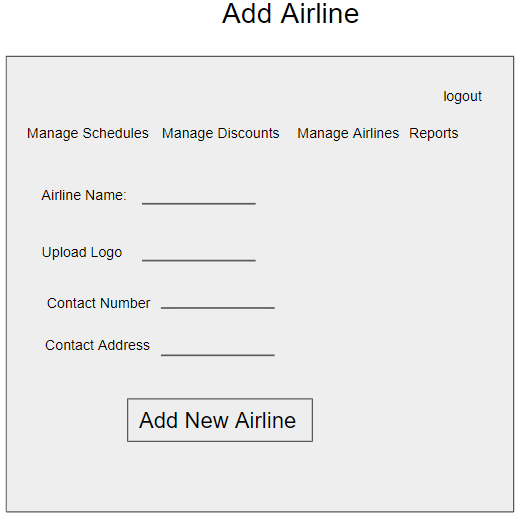
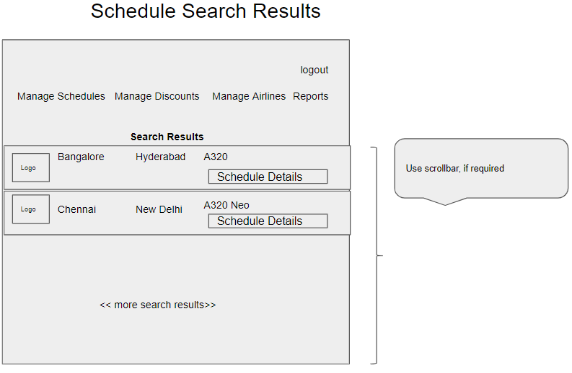
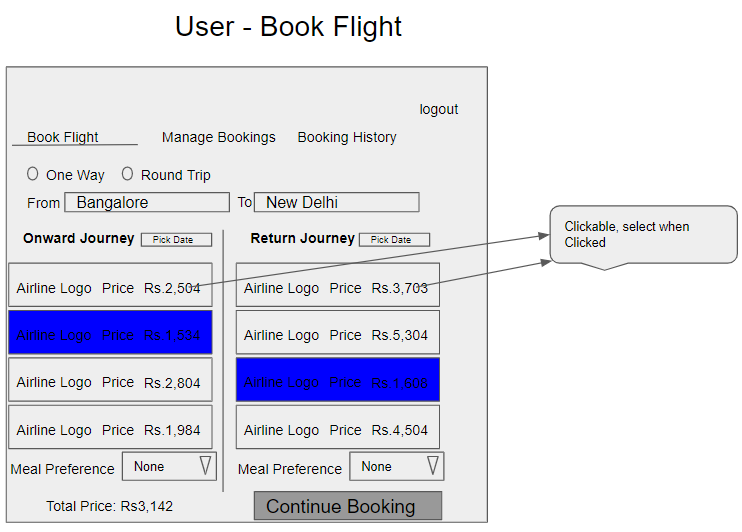
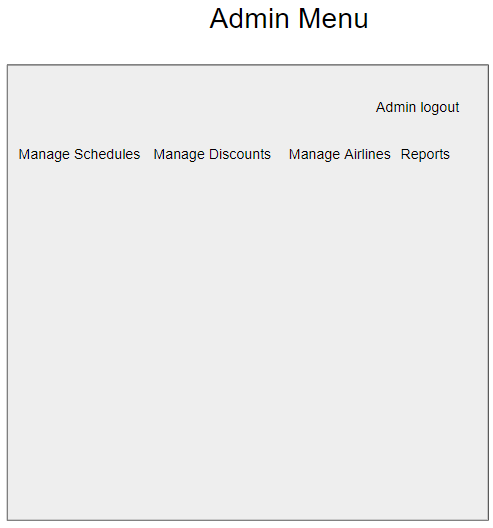
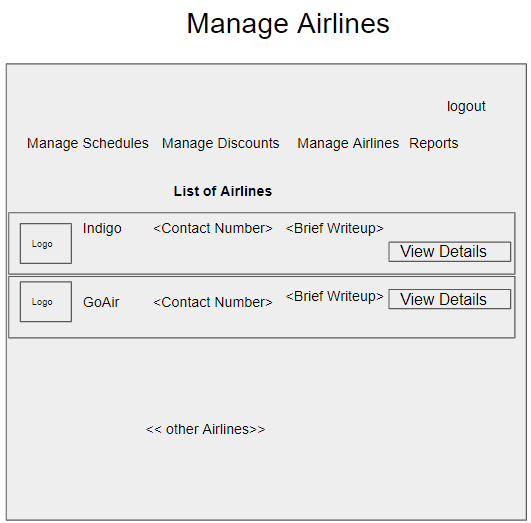
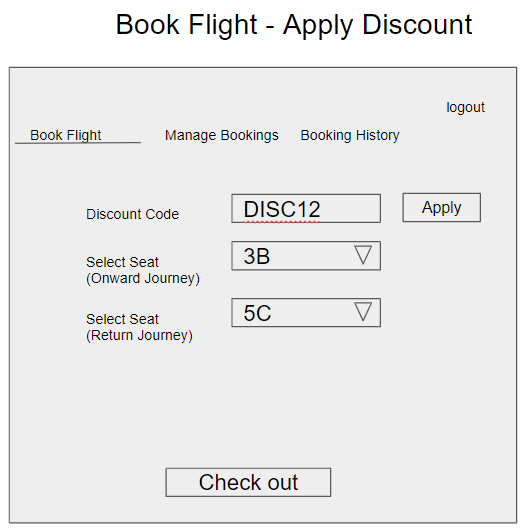
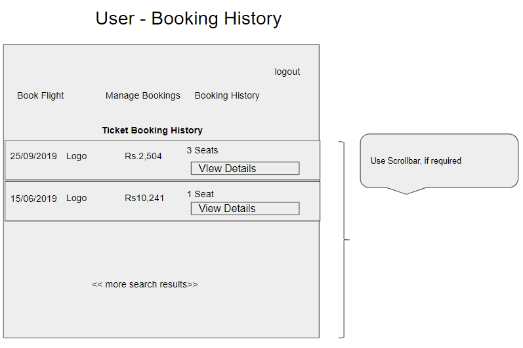
# Problem Statement

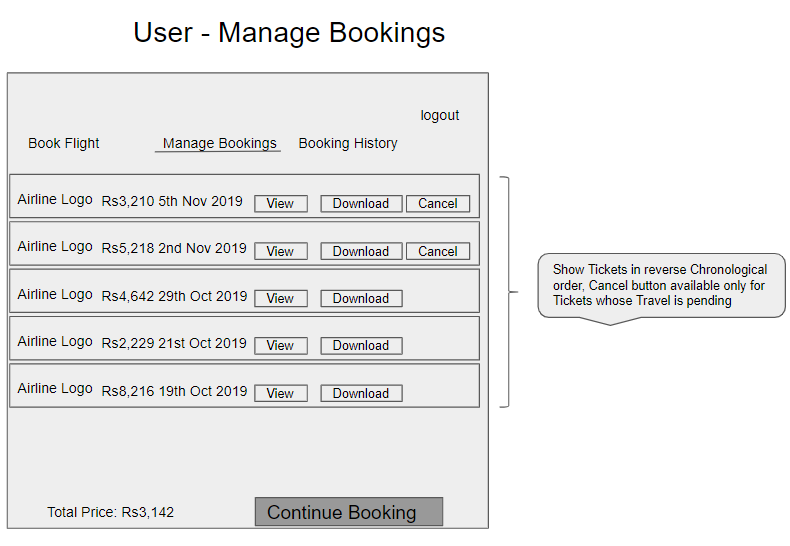
**Build a software system which lets user search for a Flight Ticket and book it & includes Admin related activities. User can also cancel or update the Ticket. Below are the different roles, which need to be supported by above Software System.**

1. **User**
2. **Admin**

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

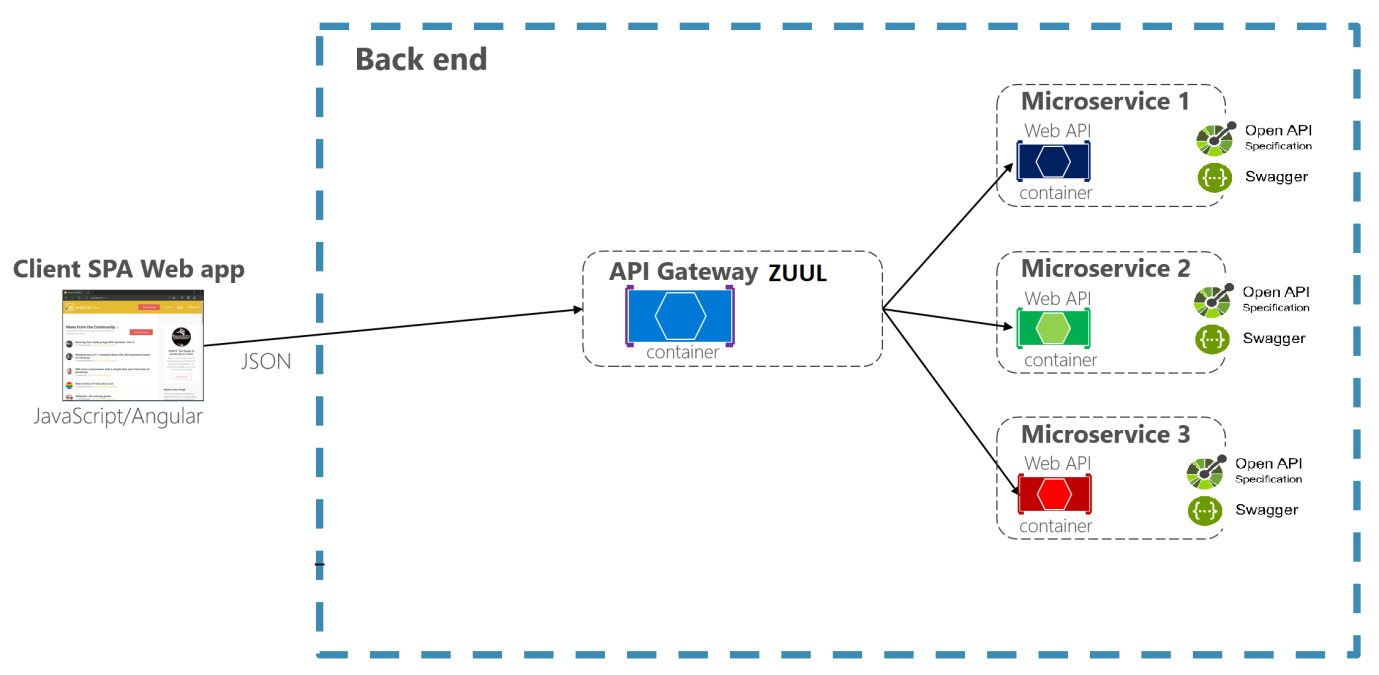
# WIREFRAMES

**UI needs improvisation and modification as per given use case.**



# Application Architecture

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

****

# 

# Tool Chain

| **Competency** | **Skill** | **Skill Detail** |
| --- | --- | --- |
| **Engineering Mindset** | **Networking and Content Delivery** |  |
|  | **DevOps** |  |
|  | **Secure Coding** |  |
|  | **Code Quality** | **Sonar** |
| **Programming Languages** | **Application Language** | **Java** |
| **Products & Frameworks** | **Presentation** | **Angular** |
|  |  | **Material/Bootstrap, rxjs, ngrx/store** |
|  |  | **Javascript/Typescript** |
|  | **Networking and Content Delivery** | **Api gateway** |
|  | **Security and Identity** | **OpenIAM** |
|  | **Compute & Integration** | **Spring Boot** |
|  |  | **Kafka** |
|  | **Database & Storage** | **MySQL** |
|  | **Governance & Tooling** | **Git** |
|  |  | **Junit** |
|  |  | **Mockito** |
| **Engineering Quality** |  |  |
| **Platform** | **Cloud Tools** |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# 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** |

# 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**