ARS-V3 (Airline Reservation System-version3.0 – Microservice based application)

* ARS-v3
  + Backend
    - Discovery Server [Eureka/Consul/Zoo keeper]
    - API Gateway (Routing to the respective microservice from a central point)
    - Config Server (Optional)
    - Microservice-1 (user-service) [each service in a different git repo]
    - Microservice-2 (booking-service)
    - Microservice-3 (payment-service)
    - Microservice-4 (rating-service)
    - Microservice-5 (offer-service)
    - Docker-compose.yml
    - K8s
      * deployment.yml
  + Frontend
    - React/nextjs project (micro-frontend)
  + Readme.MD
  + Artifacts ( HLD, LLD, Diagrams)

Application Workflow diagram

1. Usecase-1 User booking a flight ticket
   1. User needs to login in the application with valid credentials (Authenticate the user and store the JWT in session storage) [user-service & auth-service – api-gateway]
   2. Search for flight [flight-service] source & destination places/airports and date, no of passengers.
   3. Select the suitable flight-service and proceed to booking, [needs to fill all passenger details]
   4. Click on book or pay-now button. Simulate payment using mock service.
   5. Based on the status of payment, update booking details.

Service – to – Service Communication -- Microservice Communication.

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-openfeign</artifactId>

</dependency>

1. Sync Communication (Blocking) – REST Template, FeignClient
2. Async Communication (Non-blocking) [Event Driven Communication RabbitMQ/Kafka]

Open Feign -Declarative REST client.

Testing service end-points

1. Using browser (get end-points) – Path variable
2. Using Postman/Soap UI/Bruno/Swagger (Http/Web Clients)