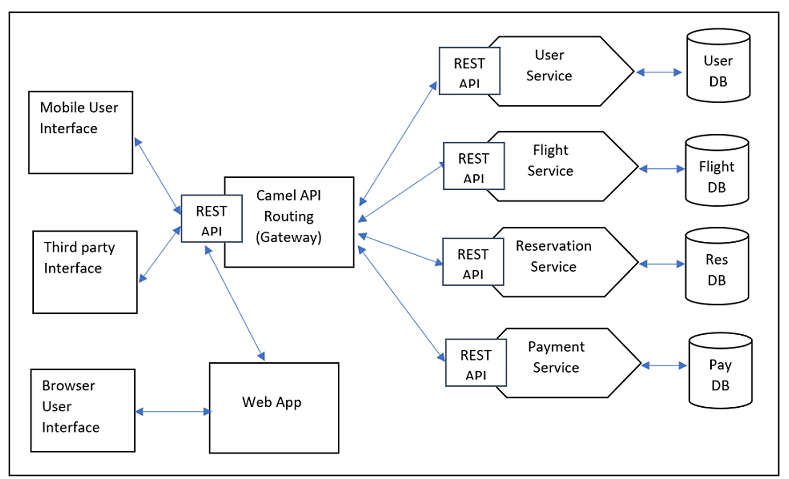
**Microservices for the airline system:**

***Architecture diagram:***

The below diagram represents on high level how the different components in the system interact with each other. User, Flight, Reservation and Payment micro services were shown in the diagram.

****

**Notes:**

* Camel gateway will be the routing gateway in the system.
* Camel routes request to the Rest API end points across different microservices.
* For calls from the browser interface, the web app interacts with Camel gateway to access the REST API.
* Inter-communication between microservices also uses Camel API routing.
* Each microservice talks to only its database.
* These are plain microservices not using service discovery function.
* In production usually each microservices are deployed on multiple nodes and Docker, Kubernetes orchestration will be used for load balancing purpose.

***Microservices:***

On high level I took four services i.e., User, Flight, Reservation and Payment services to explain their functions with some implementation details and how they fit in the overall design.

I have explained some details below about each service.

Please note that I have mentioned below about Audit service (for auditing user actions), Communication service (to publish messages to users) because they are associated services.

**User service:**

|  |  |  |
| --- | --- | --- |
| **Name:**  User service | **Description:**  The service maintains system user data, login details, provides authentication and authorization service, | |
| **Implementation:** The service will be implemented in Spring Boot rest service. The routing happens through Camel API gateway. | | |
| **Operations:**  - Register user details  - Update user details  - Authenticate and authorize user  - Verify user profile  - Assign system users to different groups and roles | | **Interfaces:**  - Queries database  - AMQ publishing  **Dependencies:**  - Audit service to audit user actions  - Communication service to publish |

**Data model:**

User service access user database which will have the following tables. Please note “USR” is schema name.

1. USR.USER
2. USR.USER\_PROFILE
3. USR.GROUP
4. USR.ROLE
5. USR.USER\_GROUP
6. USR.USER\_ROLE
7. USR.USER\_LOGIN\_DETAIL

**Flight service:**

|  |  |  |
| --- | --- | --- |
| **Name:**  Flight Service | **Description:**  This services stores and provides flight details, check flight availability and type of flight based on route | |
| **Implementation:** The service will be implemented in Spring Boot rest service. The routing happens through Camel API gateway. | | |
| **Operations:**  - Add flight details  - Manage flight details  - Get flight details  - Check flight availability | | **Interface:**  - Queries database  **Dependencies:**  - Reservation service  - Audit service |

**Data model:**

The following tables will be part of flight service. Please note that “FL” is the schema name.

1. FL.FLIGHT
2. FL.FLIGHT\_DETAIL
3. FL.FLIGHT\_SCHEDULE
4. FL.FLIGHT\_MAINTENANCE\_DETAIL
5. FL.FLIGHT\_ROUTE\_MAP
6. Fl.FLIGHT\_TRIP\_HISTORY
7. FL.FLIGHT\_OBSERVATIONS

**Reservation service:**

|  |  |  |
| --- | --- | --- |
| **Name:**  Reservation Service | **Description:**  This service provides functions to create reservation booking, ticket generation, invoking payments processing and publishing user alerts. | |
| **Implementation notes:** The service will be implemented in Spring Boot rest service. The routing happens through Camel API gateway. To sent messages to users through mobile alerts the AMQ asynchronous messaging can be used. | | |
| **Operations:**  - Create booking  - Issue ticket  - Get booking details  - Manage booking  - Provide travel preferences including meal and seat location and others.  -Rescheduling  - Ticket cancellation  - Publish alerts to user about booking details, travel alerts, through | | **External components:**  -Queries database  -AMQ publishing  **Dependencies:**  - User service  - Audit service  - Flight service  - Payment service  - Communication service (for user notifications) |

Data model:

1. RES.BOOKING
2. RES.BOOKING\_DETAIL
3. RES.TICKET\_DETAIL
4. RES.BOOKING\_USER\_PREFERENCES
5. RES.BOOKING\_USER\_NOTES
6. RES.BOOKING\_USER\_ALERTS

**Payment service:**

|  |  |
| --- | --- |
| **Name**:  Payment Service | **Description**:  The service is used to process billing for the booking. It accepts credit card details and booking information, processes the payment and issue alerts to users about payment. |
| **Implementation notes:** The service will be implemented in Spring Boot rest service. The routing happens through Camel API gateway. To send messages to users through mobile alerts the AMQ asynchronous messaging can be used. | |
| **Operations:**  -Process payment  -Get payment details  -Process credit card transaction  -Get credit card info | **Dependencies:**  -Queries database  -Publishing AMQ  **Interface:**  -Audit service  -Communication service |

Data model:

1. PAY.PAYMENT\_DETAIL
2. PAY.PAYMENT\_CREDIT\_CARD\_INFO
3. PAY.PAYMENT\_USER\_ALERTS
4. PAY.PAYMENT\_BOOKING\_MAP