SEAT BOOKING SYSTEM (Office Nest)

Group 2

- Abhineet Kelley (Backend)
- Mohammad Ali Hassan (Backend)
- Sukhad Sharma (Frontend)

Roles

Abhineet Kelley (Backend)

- Database planning
- Authentication(EP1-OSB1 and EP1-OSB2)
- User Profile Management (EP1-OSB3)
- User Bookings Page (EP3-OSB1 and EP3-OSB2)

Mohammad Ali Hassan (Backend)

- API planning
- Admin Dashboard (EP2-OSB1 and EP2-OSB3)
- Manage My Infrastructure (EP2-OSB2)
- Bookings Approval (EP2-OSB4)
- Seat Swap approval(EP3-OSB3)

Sukhad Sharma (Frontend)

- UI and App Flow
- User Bookings Page UI (EP3-OSB2)
- User Profile UI (EP1-OSB3)
- User Registration and Login UI (EP1-OSB1, EP1-OSB2 and EP2-OSB1)
- Admin Dashboard UI (EP2-OSB1 and EP2-OSB3)
- Infrastructure Management System UI (EP2-OSB2)

Common

- Features Planning
- Seat Swap Request (EP3-OSB3)

Major Features of the Application

SEAT FEATURES

Seat location

- Office (tech hub,HQ)
- Floor (1st, 2nd, 3rd)
- Seat (unique seat number for each floor)
- Coordinates:
    - Office-Floor-Seat
    - Example: HQ-2-B3

USER FEATURES

Register

- verify email via OTP
- validate inputs
- register in database

Login

-login with registered email and password
- validate inputs

Booking System

- Book a seat with a schedule:
    - Check if the seat is not taken
    - Start Time, End Time
- Search for a seat:
    - Office, Floor, Seat
    - Green: booked, Grey: unavailable
- No collisions with another booking schedule

Profile

- Edit his details
- Change password
- Logout

Dashboard

- Current datetime visible in the navbar for convenience
- Send a cancellation request to admin
- View pending cancellation requests
- View pending swap requests
- View my bookings

ADMIN FEATURES

Login (same)
Register (N/A)

Booking system

- Accept/reject booking request

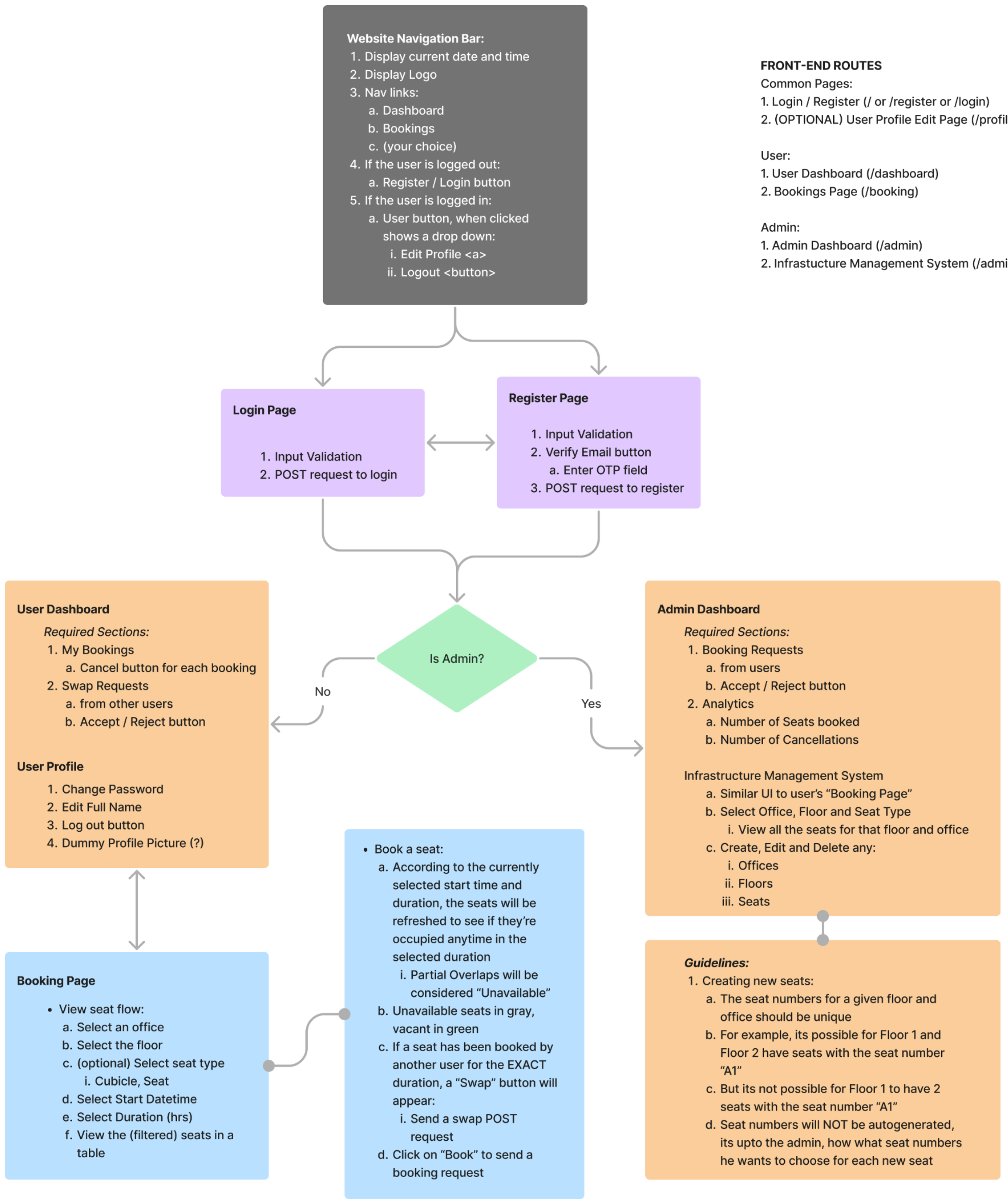
Seat Management

- Add/Remove/Edit Locations/Floors/Seats
    - If removing a seat that’s already booked, give a warning to admin

FRONT-END ROUTES
Common Pages:
1. Login / Register (/ or /register or /login)
2. (OPTIONAL) User Profile Edit Page (/profile)

User:
1. User Dashboard (/dashboard)
2. Bookings Page (/booking)

Admin:
1. Admin Dashboard (/admin)
2. Infrastucture Management System (/admin/manage)



Website Navigation Bar:
1. Display current date and time
2. Display Logo
3. Nav links:
    a. Dashboard
    b. Bookings
    c. (your choice)
4. If the user is logged out:
    a. Register / Login button
5. If the user is logged in:
    a. User button, when clicked shows a drop down:
        i. Edit Profile <a>
        ii. Logout <button>

Login Page
1. Input Validation
2. POST request to login

Register Page
1. Input Validation
2. Verify Email button
    a. Enter OTP field
3. POST request to register

User Dashboard

Required Sections:
1. My Bookings
    a. Cancel button for each booking
2. Swap Requests
    a. from other users
    b. Accept / Reject button

User Profile
1. Change Password
2. Edit Full Name
3. Log out button
4. Dummy Profile Picture (?)

Booking Page
- View seat flow:
    a. Select an office
    b. Select the floor
    c. (optional) Select seat type
        i. Cubicle, Seat
    d. Select Start Datetime
    e. Select Duration (hrs)
    f. View the (filtered) seats in a table

- Book a seat:
    a. According to the currently selected start time and duration, the seats will be refreshed to see if they're occupied anytime in the selected duration
        i. Partial Overlaps will be considered "Unavailable"
    b. Unavailable seats in gray, vacant in green
    c. If a seat has been booked by another user for the EXACT duration, a "Swap" button will appear:
        i. Send a swap POST request
    d. Click on "Book" to send a booking request

Admin Dashboard

Required Sections:
1. Booking Requests
    a. from users
    b. Accept / Reject button
2. Analytics
    a. Number of Seats booked
    b. Number of Cancellations

Infrastructure Management System
    a. Similar UI to user's "Booking Page"
    b. Select Office, Floor and Seat Type
        i. View all the seats for that floor and office
    c. Create, Edit and Delete any:
        i. Offices
        ii. Floors
        iii. Seats

Guidelines:
1. Creating new seats:
    a. The seat numbers for a given floor and office should be unique
    b. For example, its possible for Floor 1 and Floor 2 have seats with the seat number "A1"
    c. But its not possible for Floor 1 to have 2 seats with the seat number "A1"
    d. Seat numbers will NOT be autogenerated, its upto the admin, how what seat numbers he wants to choose for each new seat

DATABASE

1. User
    a. user_id SERIAL PRIMARY KEY
    b. user_fullname VARCHAR
    c. user_email VARCHAR UNIQUE
    d. user_password (hashed) VARCHAR(70)
    e. user_roles VARCHAR { comma separated roles in a string }

2. Office
    a. office_id SERIAL PRIMARY KEY
    b. office_name VARCHAR

3. Floor
    a. floor_id SERIAL PRIMARY KEY
    b. floor_number INTEGER
    c. floor_office_id Office FOREIGN KEY

4. Seat
    a. seat_id SERIAL PRIMARY KEY
    b. seat_number VARCHAR
    c. seat_type VARCHAR ("SEAT", "CUBICLE")
    d. seat_floor_id Floor FOREIGN KEY
    e. seat_booked BOOLEAN

5. Booking
    a. booking_id
    b. booking_user_id User FOREIGN KEY
    c. booking_floor_id Floor FOREIGN KEY
    d. booking_seat_id Seat FOREIGN KEY
    e. booking_start_datetime
    f. booking_end_datetime
    g. booking_status { PENDING, ACCEPTED, REJECTED }

6. (Swap)Request
    a. request_id
    b. request_booking_1_id FOREIGN KEY
    c. request_booking_2_id FOREIGN KEY

    d. request_status { PENDING, ACCEPTED, REJECTED }

7. Cancellations
    a. cancellation_id
    b. cancellation_user_id FOREIGN KEY
    c. cancellation_booking_id FOREIGN KEY
    d. cancellation_status { PENDING, ACCEPTED, REJECTED }

API ENDPOINTS

Landing/Login/Register Page

- POST /register
    - Request:
        - user_fullname (String)
        - user_email (String)
        - user_password (String)
        - user_otp (String)
    - Reponse
        - data
            - user_id
            - user_fullname
            - user_email
            - user_roles
        - token
        - success: true

- POST /verify-email
    - Request:
        - email (String)
    - Response:
        - true (OTP sent successfully)
        - false (Some error occurred)

- POST /login
    - Request:
        - user_email
        - user_password
    - Reponse
        - data
            - user_id
            - user_fullname
            - user_email
            - user_roles
        - token
        - success: true

- POST /autologin
    - Headers:
        - Authorization = token (String)
    - Reponse
        - data
            - user_id
            - user_fullname
            - user_email
            - user_roles
        - token
        - success: true

========================

User Dashboard

- GET /user/incoming-swap-requests
    - Headers:
        - Authorization = token (String)
    - Response:
        - data
            - list of Swap Requests
        - success: true or false

- POST /user/modify-swap-request //accept or reject swap request
    - Headers:
        - Authorization = token(String)
    - Request:
        - Request_id
        - Accepted:true or false
    - Response
        - Success:(true or false)

- GET /user/bookings
    - Headers:
        - Authorization = token (String)
    - Response:
        - data:
            - list of Booking objects
        - success: true

- GET /user/cancellations
    - Headers:
        - Authorization = token (String)
    - Response:
        - data:
            - list of Cancellation objects
        - success: true

- POST /user/cancellation //sends new cancellation request
    - Headers:
        - Authorization = token (String)
    - Request:
        - booking_id
    - Response:
        - data:
            - Cancellation object
        - success: true

=======================

User Profile Page

- PATCH /user/change-password
    - Headers:
        - Authorization = token (String)
    - Request:
        - user_old_password
        - user_new_password
    - Response (correct old password)
        - success: true
    - Response (wrong old password)
        - success: false

- PATCH /user/edit-details [NAME for now]
    - Headers:
        - Authorization = token (String)
    - Request:
        - user_fullname
    - Response:
        - success: true or false

=======================
Booking Page

- GET /offices
  - Headers:
    - Authorization = token (String)
  - Response:
    - data:
      - list of Office objects
    - success: true

- GET /floors/{office_id}
  - URL Parameters:
    - office_id
  - Headers:
    - Authorization = token (String)
  - Response:
    - data:
      - list of Floor objects
    - success: true

- GET /seats/{floor_id} [Dependent on prev 2 reqs, officeID, floor ID]
  - Headers:
    - Authorization = token (String)
  - URL Parameters:
    - floor_id
  - Response:
    - data
      - list of Seat objects
    - success: true

- GET/seat/details/{floor_id}
  - Headers:
    - Authorization = token (String)

    - URL Parameters:
        - floor_id
    -
    - Response:
        - data
            - list of Booking objects
        - success: true

- POST /seat/book [Can book multiple seats for now]
    - Headers:
        - Authorization = token (String)
    - Request:
        - floor_id
        - seat_id
        - start_datetime
        - end_datetime
    - Response:
        - data
            - Booking object
        - success: true

- POST /seat/swap
    - Headers
        - Authorization = token (String)
        -
    - Request:
        - Data
            - Booking object
    - Response
        - Data
            - Swap request object (status:pending)
        - success:true

========================

Admin Dashboard

- GET /bookings (status: pending)
    - Headers:
        - Authorization = token (String)
    - Response:
        - data
            - list of Booking objects with booking_status: PENDING
        - success: true

- PATCH/modify-booking
    - Headers:
        - Authorization = token (String)
    - Request:
        - booking_id
        - accepted: true or false
    - Response:
        - data
            - Booking object
        - success: true

- GET /cancellations
    - Headers **
    - Response:
        - data
            - list of Cancellations
        - success: true

- POST /modify-cancellation
    - Headers **
    - Request:
        - cancellation_id Cancellation
        - accepted: true or false

========================

IMS (Infrastructure Management System)

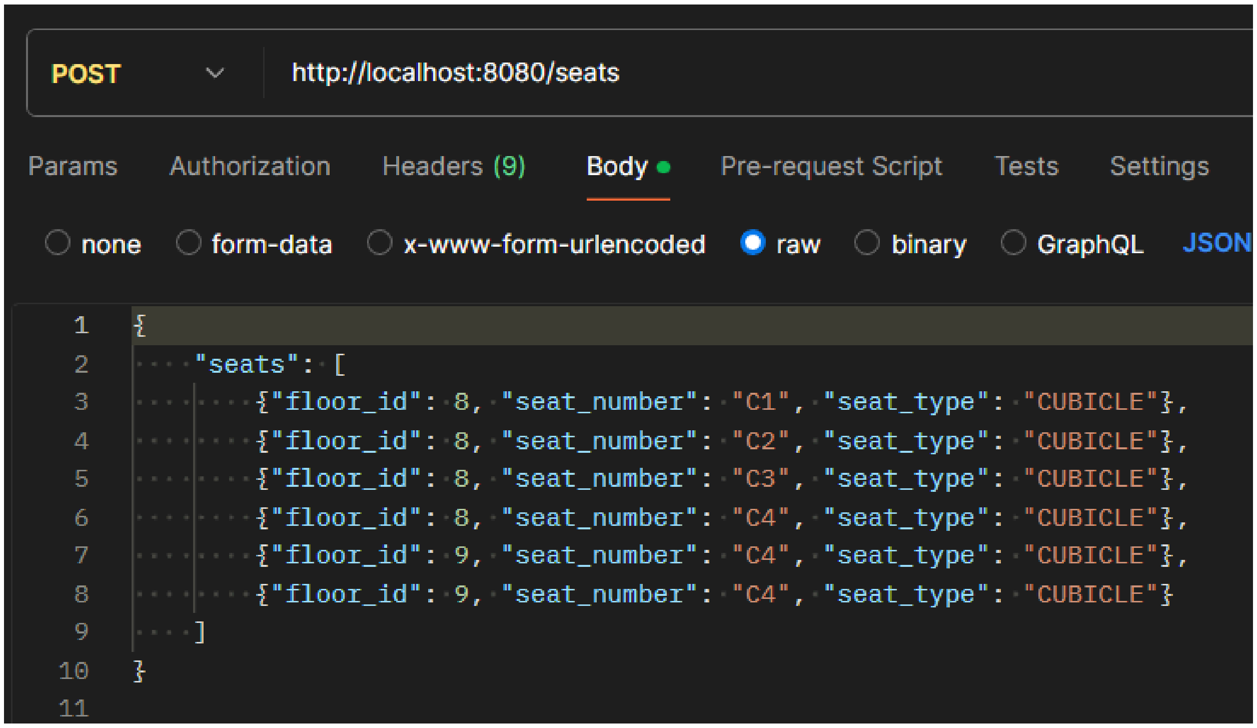
(GET offices, floors/office-id, POST get-seats from before)

- POST /office?office_name=OFFICE_NAME
    - Headers:
        - Authorization = token (String)
    - URL Parameter:
        - office_name (String)
    - Response:
        - data
            - Office object
        - success: true

- POST /floor?floor_number=NUMBER&office_id=OFFICE_ID
    - Headers:
        - Authorization = token (String)
    - URL Parameters:
        - office_id
        - floor_number
    - Response:
        - data
            - Floor object
        - success: true

- POST /seats (Create multiple seats for a floor ID)
    - Headers:
        - Authorization = token (String)
    - Request:
        - seats
            - list of Seat objects
                - Each Seat object contains:
                    - floor_id ->Floor
                    - seat_number ->String
                    - seat_type: CUBICLE or SEAT

    Response:
        - success: true



```
{
    "seats": [
        {"floor_id": 8, "seat_number": "C1", "seat_type": "CUBICLE"},
        {"floor_id": 8, "seat_number": "C2", "seat_type": "CUBICLE"},
        {"floor_id": 8, "seat_number": "C3", "seat_type": "CUBICLE"},
        {"floor_id": 8, "seat_number": "C4", "seat_type": "CUBICLE"},
        {"floor_id": 9, "seat_number": "C4", "seat_type": "CUBICLE"},
        {"floor_id": 9, "seat_number": "C4", "seat_type": "CUBICLE"}
    ]
}
```