Project brief

Intro

The purpose of the project is to build a simplified digital services ecosystem for a digital nomad - a person who travels around the world and works remotely. In our scenario we assume that such a person is going to need a place of stay (a flat or a house), a place to work (office, coworking space), a car and a parking spot.

Each Lab group will be divided into 4 teams:

- Carly
- Flatly
- Parkly
- Officely (we're waiting for a better name proposal from you…)

Each team (Carly, Flatly, Parkly, Officely) delivers:

- Admin Web App to manage their operations
- Mobile App for end users to
    - book items from their own offering
    - book items from the offering from their partner
- Backend APIs to:
    - support own operations
    - allow partner to book items via API

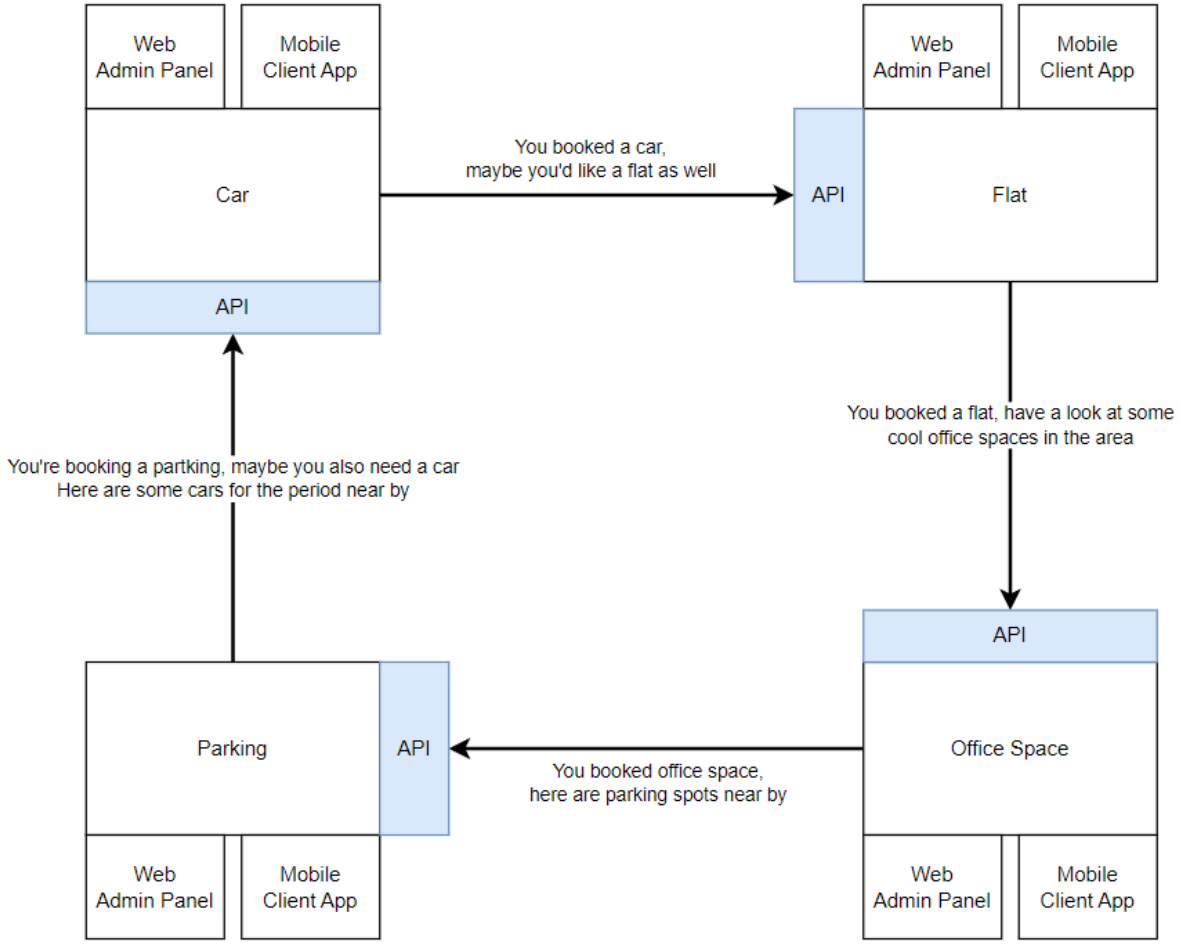
For instance (the example uses Officely and Parkly):

A user installs the officely mobile app and looks for the office in Warsaw for a month between December 1 and December 31 2023. After finding the right offering, the user is booking the specific office for that period of time. The app then asks, "hey since you have rented the office maybe you'll be interested in renting the parking space. Then the app offers parking spaces that are in the proximity of 1km from the office that was rented. Now user can choose one of the offerings and book that parking spot as well.
Note that in this example it is the Mobile App owned by Officely so the office is booked from their own system but the parking space is booked in the Parkly system.
So now if:

- admin of Officely opens the admin website of officely - then he/she will see that the booking for an office was made
- **but also** when and admin of Parkly opens the admin website of parkly - then he/she will see that the booking for a parking spot was made as well

The diagram below illustrates the idea of:

- Carly mobile app allowing to book **cars** but also **flats**
- Flatly mobile app allowing to book **flats** but also **office spaces**
- Officely mobile app allowing to book **office spaces** but also **parking spots**
- Parkly mobile app allowsing to book **parking spots** but also **cars**



Detailed requirements

Each team should build **at least** the following capabilities into their systems.

Admin Web App

ReactJs - based web application for system administrators with the following capabilities:

- Panel for viewing, adding, removing, editing items.
- Panel for viewing existing items booking, including info on the:
    - booking period
    - which system has made the booking
    - details of a user who made the booking
    - (maybe) ability to cancel the booking
- All views/panels/tables that contain multiple ‘rows’ of data should implement the **server side** search, sorting and paging.
- Among other relevant metadata it should be possible manage image/images of rented item, it’s location,

End-User Mobile App

React Native - based mobile application for end users (customers) with the following capabilities:

- Ability to see the list of available items from their own system (eq. if you are a Carly then you should be able to see, and search for cars)
- Ability to search for items to rent/book (in own system) by at least:
    - name
    - location
    - proximity from the given location (ex. show me all flats within 5km from the given coordinates)
    - other params (for each business case params are different, like for cars you might look for number of seats, brand, model, etc. while in flats you search for a number of rooms, and floor etc.)
- You should only see items that are available based on renting period you selected
- You should only see items price calculated for the selected period
- Ability to **book** selected item
- Ability to search for items to rent/book in your partner system.
- You should only see partner items that are available based on renting period you selected
- You should only see partner items price calculated for the selected period (calculation should come from the partner)
- Ability to see all your bookings
- Ability to cancel your bookings (if you used Carly mobile app to book a flat then you should be able to cancel that booking (of a flat), without the same app

REST APIs

APIs that you build should primarily focus on supporting business requirements defined above.

Other requirements

- All users should be authenticated and visibility of items need to be adjusted accordingly, for example: end-user who has booked one apartment can only see his own booking and not all bookings in the system
- Part of the submission will be the API documentation (details can be discussed when we reach backend classes)
- Part of the submission will be UX Design (Primarily Wireframes)

Interesting problems, considerations to be discussed along the semester:

- It is quite natural that a person who made a booking, has decided to cancel it. It is also relatively easy to assure that our system allows that. Consider however a scenario where a user has used the mobile app of **Parkly** to book a **car.** Then the car breaks down before the rental period and the company owning the car (Carly) needs to cancel the booking. What are the technical and UX challenges here?

- Design choice:
    - mobile and web app of one system talking directly to API of the other system vs
    - mobile and web app talking **only** to API of own system (communication only ‘backend-backend’)

  Consider pros and cons of both approaches.