# Week 4 – Research – Manjyot Joher (12897981)

## Topics:

* React native virtual testing environment
* Relevant APIs
* Tentative MongoDB schema
* Google OAuth

### React native virtual testing environment

React native has multiple mobile simulators, with the default being iPhone (iPhone 11). After the code and tech stack has been setup and completed, the simulator can be launched by using the npx command: *npx react-native run-ios*. As react native doesn’t have a built-in executable simulator for android, a physical android device or the android studio simulator would need to be used. Similarly, apps can also be executed using the Expo Go app on both mobile platforms for easier ‘real-world’ testing.

### Relevant APIs

Trivago:

* Trivago’s express booking API contains the following methods. These methods could be used to gather data to populate Travelogue or verify booking availabilities if decided as a part of extra app features during later sprints or the second semester.
  + Booking\_availability: checks booking availabilities
  + Booking\_prepare: Specifies if the current booking rate is still open
  + Booking\_submit: submits a room booking
  + Booking\_payment\_authorise: authorises a booking via a 3-Domain service or Paypal
  + Booking\_verify: verifies booking status

Other travel providers such as Expedia, Booking.com and TripAdvisor have similar API endpoints but require an approved application for their partner program, which may not be guaranteed for a student project.

Stripe:

* The Stripe payment processing API could be utilised to demonstrate that the app is able to charge fees from real bank accounts when making a ‘booking’. Depending on time constraints and privacy policies, this could be implemented into the final app to demonstrate a working payment system without confirming a real booking for hundreds of dollars.

Google Maps API:

* The Google Maps API will be utilised to display interactive maps of booked and nearby available rooms.

### Tentative MongoDB schema

Graphical user interface

Description automatically generatedThe schema below contains 4 key collections that represent core objects in the application. These are Hotels, Rooms, Users and Bookings. This can be adapted to include more collections in regard to ML recommendations, as well as the modification of the User collection. The schema notations can also be representative of arrays within each collection. For example, each Hotel can contain an array of Rooms, while each Room can have an array of Bookings, and each Booking, can have an array of Users.

### Google OAuth

Based on recent team discussions, it was decided that we would implement Google OAuth, so we wouldn’t need to handle the authentication locally. This was compared to PassportJS, which can enable a customised token / session-based authentication service; however, we decided to try something new and straight forward to integrate (Google OAuth).

<https://developers.google.com/identity/protocols/oauth2>