Trippy - We're not airbnb!

In this project we'll create an airbnb like web site, the core functionality of the site will be:

1. Show locations (places you can rent) and allow to search them
2. User authentication (login/logout and sign up)
3. Book a location

General notes

1. The client and server will be two separate projects.
2. Use pure css (no frameworks)
3. Use flexbox and/or CSS grids for layouting
4. The UI/UX should look good, treat it as though real people will be using it
5. Responsiveness. The app should support these resolutions:
    a. Mobile - up to 768px
    b. Tablet - from 769px up to 1024px
    c. Desktop - from 1025px

Milestones

Milestone 1 - Location page (6 hours)

1. Create a new app and create the location page. Here's an example page to get the idea: https://www.airbnb.com/rooms/12522.
   Our version will be a simpler page that should contain:
    a. An image carousel
    b. All location info (see *entities* section below)
    c. A map with a pin to show the location (use google maps)
    d. Location reviews (display and creation)
    e. Booking section
2. Create a demo data set representing the location's data, for now (until we build the server) store in the client side.
3. Figure out how you're going to split this page into components. How will they interact with each other ?
4. Create those components one by one, write tests whenever necessary.

Milestone 2 - Site template and routing (2 hours)

1. Create the template of the site. All pages will have the following:
    a. Header - with logo and login/logout/signup.

    b. Main content area - changes to show the current page
2. Create stubs for the following page: Home, Location, Signup, Login and add a router

Milestone 3 - Create the availability widget (6 hours)

1. shows the room availability as little squares, one for every day of the month., a full square means that the location is booked on that date.
2. Prev/next buttons to move through months (don't allow to go to past months)
3. clicking an available date puts its value in the populates the "start date" field of the book section
4. shift clicking a later date populates the "end date" field of the book section
5. create unit tests for the widget

Milestone 4 - Home page (7 hours)

1. Create the home page for the app. It should look something like Airbnb's home page but simpler. It will contain the following elements:
    a. Search - will filter the location shown. Allow search by city, number of guests, price, date range
    b. List of locations. Each location should show its title, price, rating, image and city/country
2. Clicking a location will navigate to the location page

Milestone 5 - Basic server - Mongodb (15 hours)

1. Create a REST API server with a few basic endpoints:
    a. Get locations
2. Cover all endpoints with unit tests

Milestone 6 - More data (6 hours)

1. Implement login system
2. Cover all endpoints with unit tests

Milestone 7 - Connect the client and the server (2 hours)

1. Change the client to fetch all the data from the server. Use a proxy on the client side to overcome CORS issues.
2. Make the login/logout work with the server
3. Make the signup work with the server
4. Test everything still works using the client

Milestone 8 - Add redux (15 hours)

1. Decide which of your components will become containers
2. Manage your state using redux except when setState is still appropriate

Database Entities

1. location
    a. id
    b. address
        i. city
        ii. country
        iii. street
        iv. number
        v. lat
        vi. lng
    c. title
    d. description
    e. images
    f. maxGuests
    g. ownerId
    h. price
    i. ameneties: []
2. booking
    a. startDate
    b. endDate
    c. userId
    d. locationId
3. Review
    a. id
    b. userId
    c. locationId
    d. title
    e. name
    f. content
    g. rating (1-5)
    h. date
4. users
    a. id
    b. firstName
    c. lastName
    d. image

Client directory structure

- public
    - index.html
- app
    - components
        - MyComponent
            - MyComponent.js
            - MyComponent.css
            - MyComponent.test.js
    - containers
    - reducers
    - actions
    - consts
    - store.js
    - index.js

Server directory structure (inspired by this project)

- app
    - config
    - controllers
    - model
    - middleware
    - routes.js
    - database.js
    - server.js
    - index.js
- tests