Where to eat?

---

1. To run the application:

Instructions:

1. Navigate to folder "fecodingtest" in a terminal
2. Type in `npm install`
3. Type in `npm start`
4. If application don't open in browser, you find the application at `localhost:8000`

Dependencies:

- material-ui/core 3.9.1
- material-ui/icons 3.0.2
- styled components 4.1.3
- react-router-dom 4.3.1

API:

I added some lines in server.js to make the api work for me. This was due to CORS-conflicts in my browser Google Chrome. Following lines was added:

```
app.use(function(req, res, next) {
    res.header("Access-Control-Allow-Origin", "*");
    res.header("Access-Control-Allow-Headers", "Origin, X-
Requested-With, Content-Type, Accept");
    next();
  });
```

You start the api the same way as in the description for the assignment:

How to setup the API locally on your machine:
URL : localhost:3000
1. Install docker - https://www.docker.com/products/docker-desktop
2. Install docker-compose - https://docs.docker.com/compose/install/
3. Extract the contents of the .zip file
4. Make sure ports 3000, 27017 are not being used by other processes on your computer (in case you have a different instance of mongo or a web server running already)
5. Navigate into the unzipped folder through a terminal and run docker-compose up --build
6. The api will be available at http://localhost:3000

2. Features

Completed features:

- A view of all restaurants
- A view for each restaurant that displays more information

I wanted to show a map with all the restaurants. I tried to use OpenLayers because Goole Maps API needed a API-key. Google also had changed how to access their api. But OpenLayers had a tricky documentation so I decide to skip the map feature completely.

I had an ambition to sort/filter the restaurants in some way but then I decided to skip that part, since the application look clean as is. You get all information you need. When you visit the restaurant's detailed view you can access its webpage and you have all practical information that you need.

3. Design decisions

I made a small and easy application that presents the information in a clear way. Even though I decided not to use a map.

4. Bugs and improvements

The application runs without errors when I tested it. I have not detected any bugs at the moment.

Of course the application could do some improvements. For example:

- add a map to show location
- add a better background color or picture to the application

5. Reflection

I think the application shows a bit of my knowledge for React. I think this kind of assignments are very fun to work with and of course the applications could be improved. I am still learning React!

One bad thing is that I spend so much time to focus on how to get the api work. It was a lot of CORS-conflicts in my browser. So after about 2 days of trying to avoiding to change the server, I added the lines described in section 1. Then the api worked just fine. If I haven't spent so much time on that, my application my have looked a bit different.