**Facilities map**  
  
**Project subject:**  
Using the Google Places API (https://developers.google.com/places/javascript/?hl=ru), implement an app to find places near me. Implement various filters, add the ability to leave reviews. Mark visited places, users can see visited places of other users.  
  
**User requirements:  
Auth:** (Cookies)  
As a user, I want to login in the system.  
As a user, I want to register in the system.  
As a user, I want to login via google.  
As a user, I want to login via facebook.  
  
As a user, I want to change UI theme.  
As a user, I want to change language (RUS/ENG)  
  
**Technologies:  
FE:**HTML 5CSS (Styled components) – link is provided below  
EcmaScript/TypeScript (Optional)  
React   
Redux – thunk/saga (up to you)  
UI Component Libraries (necessarily) – Material UI/Ant/Semantic (up to you)  
Webpack  
 **BE:**Node JS  
EcmaScript/TypeScript  
Express/Koa  
MongoDB/PostgreSQL/MySql  
Mongoose/Sequelize  
  
**Pre start:**  
  
1. Create a github / bitbucket / etc repository and provide a link.  
2. Estimate each task of the project in hours before start it and provide estimated time for instance you can do it via google spreadsheets and send me a link or attach it in your created repository.  
3. After finishing all tasks provide me elapsed time for each task, if you spent time on research specify how much time has been spent such actions you can track in the spreadsheet you’ve created in the previous task.

**Guidelines**  
1. Frequent commit history.  
2. Write .jsx / .css / .sass / .js / .ts / .tsx in separate files.  
3. Prepare all needed npm tasks to run app / lints / tests / etc.  
4. Describe in readme file how to start and use your project’s tasks.  
5. Code should be readable so make sure it ts adjusted and easy to read.  
6. React is mandatory. If you need, you can use addition libraries / packages.  
7. The project should be able to run with a server that is written in node.js.  
8. Feel free to add comments in the code on possible issues / decisions you had to take.  
9. Write your solution using PropTypes/TypeScript.  
10. Use eslint. Make sure that your code without linter errors / warnings.  
11. Make the most opportunities of new EcmaScript standard if appropriate.  
12. Tests should pass.

**What else to read**  
- Styled Components <https://styled-components.com/>  
- TypeScript <https://www.typescriptlang.org/>  
- React <https://reactjs.org/>  
- Redux <https://redux.js.org/>  
- Redux thunk <https://www.npmjs.com/package/redux-thunk>  
- Redux saga <https://redux-saga.js.org/>  
- Material UI <https://material-ui.com/>  
- Ant UI <https://ant.design/>  
- Semantic UI <https://semantic-ui.com/>  
- Webpack <https://webpack.js.org/>  
- Node JS <https://nodejs.org/en/>  
- Koa <https://koajs.com/>  
- Express <https://expressjs.com/>  
- MongoDB <https://www.mongodb.com/>  
- PostgreSQL <https://www.postgresql.org/>  
- MySQL <https://www.mysql.com/>  
- Mongoose <https://mongoosejs.com/>  
- Sequelize <https://sequelize.org/>