Софийски университет “Св. Климент Охридски”

Факултет по математика и информатика

*Разработка на клиент-сървър (fullstack) приложения с Node.js + Express.js + React.js*

**Проект на тема:**

***NightLux***

Изготвили:

1. Георги Киряков, 81110

Ръководител:.......................

(Траян Илиев)

Съдържание

1. Мисия на проекта………………………………………………………...3
2. Основни потребителски случаи………………………………………..4
3. Основни изгледи………………………………………………………….5
4. Server Resource API……………………………………………………...5
5. Използвани технологии………………………………………………….7
6. Нефункционални характеристики……………………………………..7
7. Дизайн……………………………………………………………………...6
8. Заключение - срещнати/преодолени трудности и перспективи за бъдещо развитие на системата………………………………………9
9. Източници……………………………………………………………..10

Мисия на проекта

NightLux е приложение, което ще помага на хората да намират обекти и услуги, работещи в късните часове на денонощието.

Основни потребителски случаи

|  |  |  |
| --- | --- | --- |
| **Потребителски случай** | **Кратко описание** | **Участващи потребители** |
| Търсене на обекти и услуги | Потребителят може да търси обекти и услуги спрямо категория и локация. | Всички |
| Търсене на близко намиращи се обекти. | Потребителя може да търси обекти и услуги близо до неговата текуща позиция. | Всички |
| Добавяне на обект / услуга | Потребителят създава в базата данни обект / услуга като предостави категория, име, описание и локация за обекта / услугата. | Всеки |

Основни изгледи

|  |  |  |
| --- | --- | --- |
| Име на изгледа | Кратко описание | URI |
| Начало | Това е страница с поле за търсене и кратък списък от категории. | / |
| Таг / Категория | Страница, на която се показват всички обекти от дадента категория | /tags/:id |
| Обекти/Услуги | Визуализира обекти и услуги отговарящи на подадена заявка. | /services?q=заявка |
| Близо до мен | Показва обекти намиращи се близо до текущата позиция на потербителя. | /near |
| Обект/Услуга | Страница с инфорамция за обекта. | /services/:id |
| Добавяне на обект | Страница със форма за добавяне на обект. | /services/new |

Server Resource API

|  |  |  |
| --- | --- | --- |
| Име на ресурса | Кратко описание | URI |
| Услуги | services(query: String, geoQuery: String, offset: Int, limit: Int): [Service] |  |
| Услуга | service(id: ID!)  createService(input: ServiceInput): Service  deleteService(id: ID!): Service |  |
| Тагове/Категории | tags(categoriesOnly: Boolean, name: String, offset: Int, limit: Int): [Tag]системата |  |
| Таг | tag(id: ID!): Tag |  |

Използвани технологии

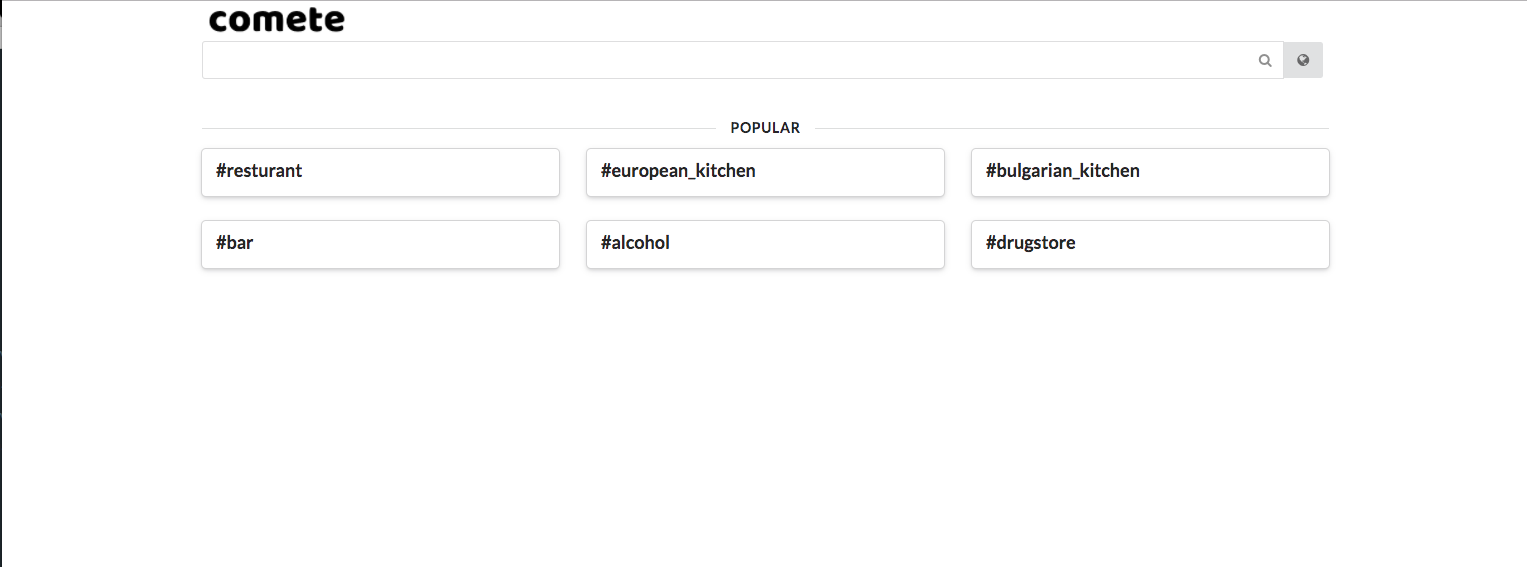
* Dev част:
  + Docker - платформа, която позволява да изграждане, доставяне и стартиране разнообразни приложения, отделяйки ги от инфраструктурата.
* Сървърна част
  + Babel - копилатор за модерен Javascript
  + Express - библиотека, предоставяща routing и middleware framework за изграждане на NodeJS приложения;
  + Mongoose - библиотека, посредством която се извършва достъп до базата данни;
  + Joi - библиотека за верификация, която се използва за проверка на потребителския вход;
  + Apollo GraphQL Server - представлява middleware, за съдаване на сървър отговарящ на GraphQL стандарта.
* Клиентска част
  + Babel - копилатор за модерен Javascript;
  + React - библиотека за реактивен UI;
  + React-router - дава възможност за предвижване между изгледи без презареждане на цялата HTML страница;
  + Apollo Client - дава възможност за свързване към Apollo GraphQL Server;
  + Lodash - библиотека, предоставяща възможност за работа с обекти, списъци, низове;
  + Google Autocomplete - предоставя API за подсказване на улични адреси;
  + Google Maps - API за географски карти
  + Semantic UI - библиотека за респонсив потребителски интерфейс

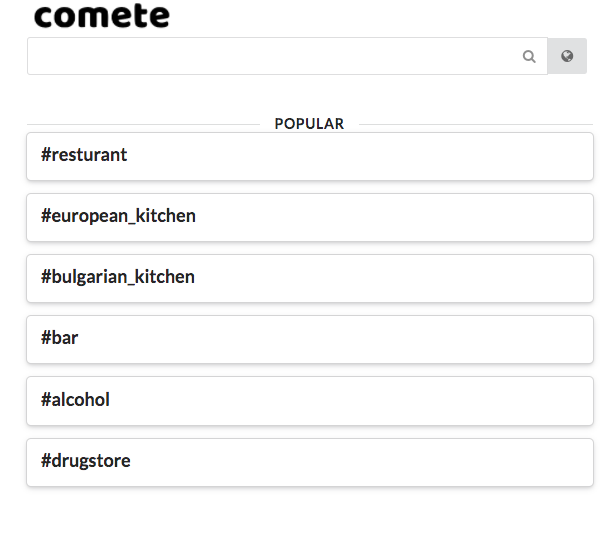
Нефункционални характеристики

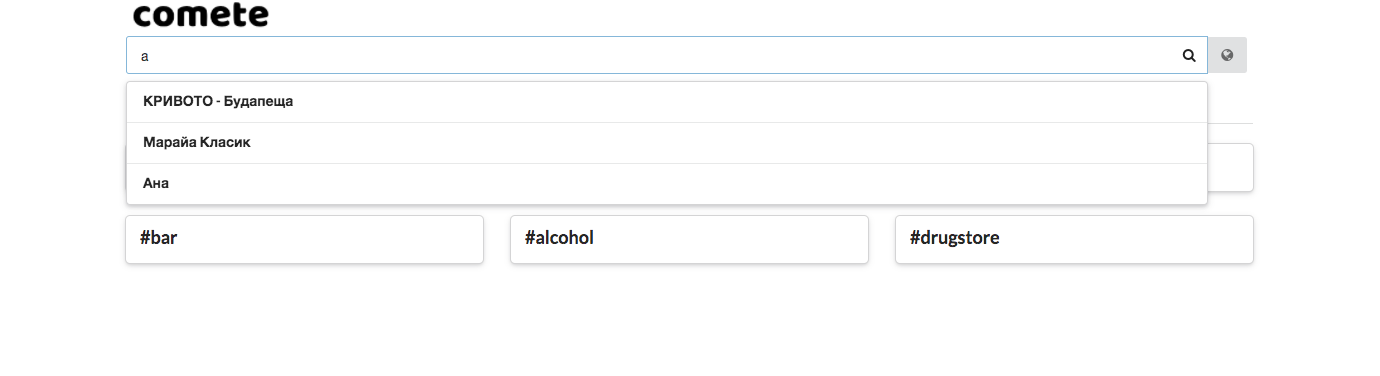
* Изчистен и лесен за разбиране дизайн.
* Свободна плафрома, на която е достъпна до всеки. Потребители няма нужда да се регистрират за да използват приложението.
* Самопопълващи се входни полета с цел улесняване на потребителя.
* При употреба от мобилно устройство с инсталирано Google Maps приложение, потребителя получва насоки до дадена услуга чрез него.

Дизайн

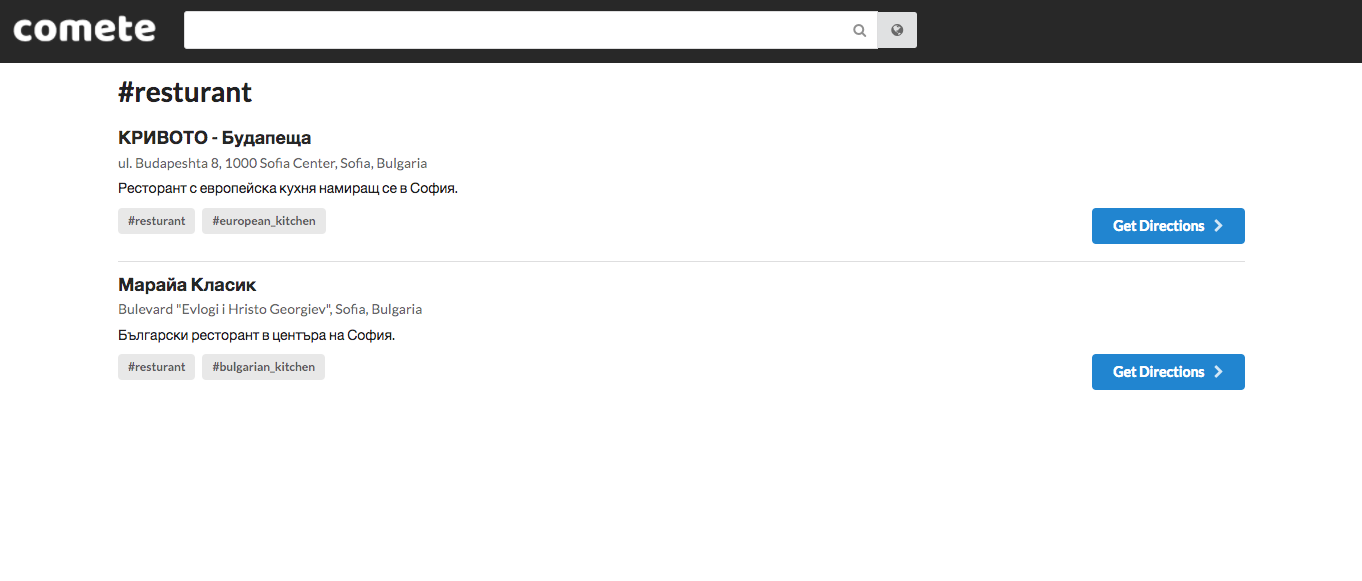
1. Начална страница

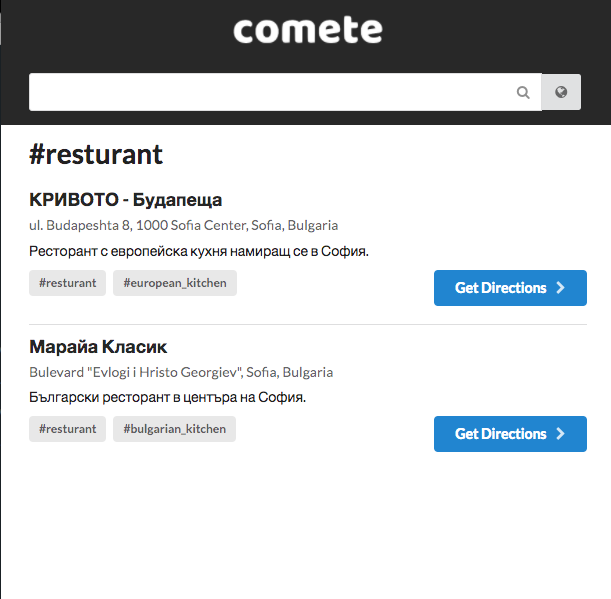




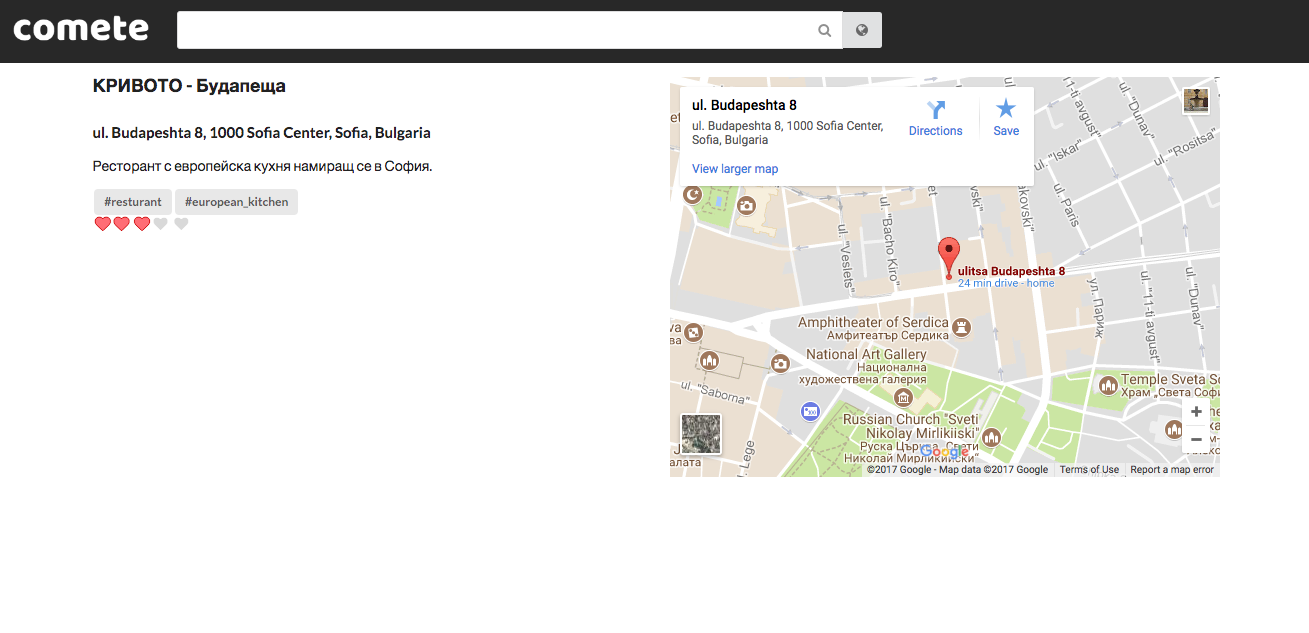


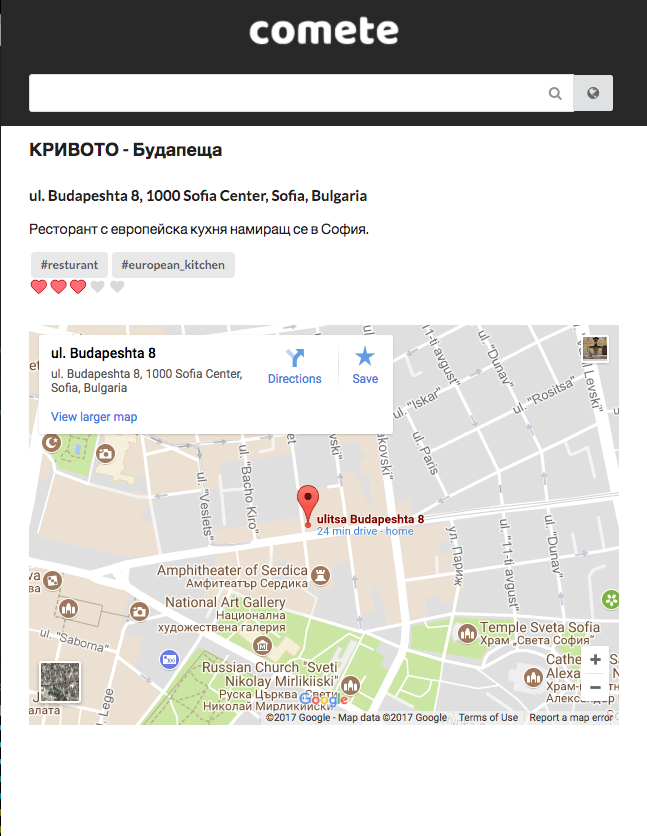
2. Категория / таг



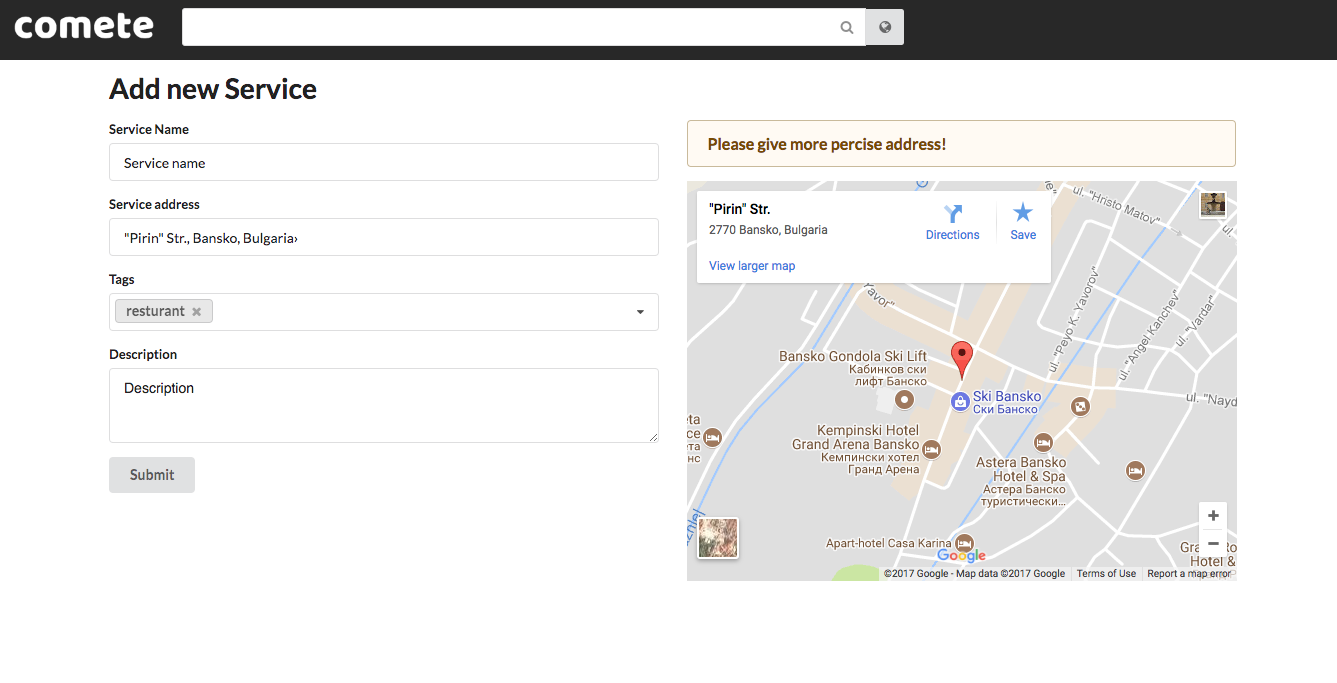


3. Услуга / Обект





4. Нова услуга/обект



Заключение - срещнати/преодолени трудности и перспективи за бъдещо развитие на системата

Проекта беше иключително интересен, тъй като с запознах с новия query език GraphQL. Основен проблем бе лисата на опит с React и GraphQL. Разбирането на това как двете технологии работят заедно отне голяма част от времето прекарано в разработване на приложението. Проблем бе и интеграция с Google Maps API, оказа се много по-сложна от колкото очаквах. За в бъдеще добавянето на снимки би подобрило потребителското преживяване. Също приложението може да се развие като място за даване на мнения за наличните обекти и услуги.

Източници

1. [APOLL] <http://dev.apollodata.com/react/index.html> - Apollo Client Documentation (август, 2017)
2. [GOAPI] Google APIs - <https://developers.google.com/> (август, 2017)
3. [GRAQL] GraphQL - <http://graphql.org/learn/> (август, 2017)
4. [REACT] <https://github.com/ReactTraining/react-router/tree/v3/docs> - React Router Documentation (август, 2017)
5. [JOIRF] <https://github.com/hapijs/joi/blob/v10.6.0/API.md> - JOI Reference (август, 2017)
6. [RCTOD] <https://facebook.github.io/react/tutorial/tutorial.html> - React Official Tutorial and Documentation (август, 2017)
7. [MNGOS] <http://mongoosejs.com/docs/api.html> - Mongoose API Documentation (август, 2017)