**Софийски университет**

**„Св. Климент Охридски”**

**Факултет по Математика и Информатика**

**Online Beauty Store**

Курсов проект

по

“ [Разработка на клиент-сървър (fullstack) приложения с Node.js + Express.js + React.js](https://learn.fmi.uni-sofia.bg/course/view.php?id=5226)”

Преподавател: Траян Илиев

30.06.2019 г.

Ваня Димитрова Любенова

ф.н. 71715

**Съдържание**

1. Увод…………………………………………………………………………………3

1.1. Описание на функционалността и не-функционалните изисквания……….......3

1.2. Описание на използваните технологии и библиотеки..……………….......…….3

2. Описание на архитектурата, реализацията и на начина на взаимодействие

между компонентите……………………………………………………………….3

2.1. Описание на архитектурно значимите интерфейси ……………………………..3

2.2. Описание на начина на инсталиране и конфигуриране………………….........…4

2.3. Потребителска документация за начина на работа със системата……….…….4

2.4. Заключение - преодолени трудности и перспективи за бъдещо развитие…..….5

3. Използвани материали……………………………………………………………...5

1. Увод

Онлайн пазаруването става все по-популярно. С няколко кликвания можете да си закупите всичко. Онлайн пазаруването придоби по-голяма популярност в сравнение с пазаруване на място. Днес, онлайн пазаруването пести време и пари. Купувачите могат да поръчват всяка стока онлайн и тя да бъде доставена до дома и офиса. Също така интернет пазаруването позволява на купувачите да сравняват цените в различни сайтове, да дават и получават бързо обратна връзка, както и да поръчват от други държави.

Онлайн магазинът за красота (OBS) предлага най-добрата дестинация за пазаруване на красота онлайн - козметични продукти, продукти за коса и грижа за кожата на най-добри цени. Освен това позволява на потребителите да се регистрират и оценяват продуктите. Системата е разработена като Single Page Application (SPA), използвайки React.js като front-end и Node.js + Express като back-end технологии. Всеки изглед има отделен URL и маршрутизирането се извършва от страна на клиента, като се използва React Router. Интерфейсът е реализиран като REST / JSON API, използвайки mongoDB. Като допълнение приложението използва PayPal Checkout чрез интеграция на Smart бутони за плащане.

Основните потребителски роли са:

• Анонимен потребител - може да преглежда само информацията за продуктите и да ги добавя в количката.

• Регистриран потребител - може да разглежда информация за продуктите, да оценява продукти, да управлява количката, да изпраща подробности за доставката и да избере да заплати кеш или чрез услугата на PayPal.

• Администратор (разширява Регистрирания потребител) - може да следи направените поръчки.

2. Описание на функционалността и не-функционалните изисквания

|  |  |  |
| --- | --- | --- |
| 1. Main Use Cases / Scenarios | | |
| **Use case name** | **Brief Descriptions** | **Actors Involved** |
| * 1. **Review products information** | The *User* can review the main products information in *OBS*, and can click on a specific product to read description. | All users |
| * 1. **Add to Cart** | *The User* can choose from the product’s page or from the product’s detail modal to add a specific item to its cart. | All users |
| * 1. **Manage Cart** | *The User* can increase/decrease each product's amount, fully empty the cart and see the tax and discount depending on the changes done. | All users |
| * 1. **Register** | *Anonymous User* can register in the system by providing a valid e-mail address, first and last name, and choosing password. | *Anonymous User,* |
| * 1. **Rate Products** | *Users can see the rating of all products. Rating is calculated using arithmetic mean of the data of all submits.*  *Registered Users can rate each product (not more than once for a product)*. | All users,  Registered Users |
| * 1. **Enter**   **Delivery Information** | *Registered Users* can add new address and telephone for their order or use the old one. | *Registered Users* |
| * 1. **Paying** | *Registered Users* can choose to pay cash or using the PayPal services. | *Registered Users* |
| * 1. **Tracking** | *Administrators* can track orders in real-time. | *Administrator* |

3. Описание на използваните технологии и библиотеки

3.1. [1] Node.js - мулти-платформена среда за изпълнение на сървърни и мрежови приложения с отворен код, която предоставя задвижвана от събития архитектура и неблокираща входно-изходна система за програмиране на приложенията (API).

3.2. [2] Express.js - framework за уеб приложения за Node.js, с отворен код, предназначенa за изграждане на уеб приложения и API.

3.3. [3] React - JavaScript библиотека за изграждане на потребителски интерфейси.

3.4. [4] Bootstrap - client-side среда с [отворен код](https://bg.wikipedia.org/wiki/%D0%9E%D1%82%D0%B2%D0%BE%D1%80%D0%B5%D0%BD_%D0%BA%D0%BE%D0%B4), която съдържа набор от инструменти за създаване на уеб приложения и уебсайтове.

3.5. [5] MongoDB - система за обработване на [бази данни](https://bg.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D0%B8) от документи, която съхранява структурираната информация в [JSON](https://bg.wikipedia.org/wiki/JSON) формат с динамични схеми.

3.6. Mongoose - MongoDB инструмент за моделиране на обекти, проектиран да работи в асинхронна среда.

3.7. nodemon - инструмент, който помага за разработване на базирани на node.js приложения чрез автоматично рестартиране на приложението.

3.8. body-parser - мидълуер модул за да обработвате на HTTP заявки.

3.9. bcrypt – библиотека за хеширане на пароли.

3.10. cors – Node.js пакет за предоставяне на Connect / Express мидълуер

3.11. jsonwebtoken – предоставя JWT кодиране и декодиране за Node.js.

3.12. axios – promise базиран HTTP клиент за браузъра и Node.js.

3.13. react-dom – лужи като входна точка за DOM и сървър render-ите за React.

3.14. react-router-dom – съдържа DOM binding-ите за React Router.

3.15. styled-components – позволява да се използва CSS код, за оформяне на компонентите.

1. Описание на архитектурата, реализацията и на начина на взаимодействие между компонентите

Back-end част на проекта:

* client (create-react-app директория)
* models (създаване на Mongoose schemas)
  + Checkout.js
  + Product.js
  + User.js
* routes (имплементиране на server endpoints)
  + Checkouts.js
  + Products.js
  + Users.js
* node-mudules
* .gitignore
* package-lock.json
* server.js (създаване на Express server, добавяне на cors и body-parser мидълуеър, конфигуриране да слуша на порт 5000 и връзване към MongoDB)
* package.json (дефиниране на потребителските команди и инсталиране на dependency-тата)

Front-end част на проекта:

* public
  + img (съхраняване на изображенията на продуктите)
  + index.html (добавяне на font библиотеките)
* node-modules
* seed
  + product-seeder.js (добавяне на информацията за продуктите в базата)
* src
  + components (създаване на компонентите на приложението)
    - Cart
      * Cart.js
      * CartColumns.js
      * CartItem.js
      * CartList.js
      * CartTotals.js
      * Checkout.js
      * EmptyCart.js
      * PayPalButton.js (интегриране на smart бутон)
    - Button.js
    - Default.js
    - Details.js
    - Login.js
    - Modal.js
    - Navbar.js
    - Product.js
    - ProductList.js
    - Profile.js
    - Register.js
    - Title.js
    - UserFunctions.js (използване на server endpoints)
  + App.css (добавяне на CSS код)
  + App.js (използване на React Fragment)
  + App.test.js
  + context.js (сетване на React Context за споделяне на данни)
  + index.css
  + index.js (използване на React Context и React Router)
  + logo.svg
  + logo\_text.svg (лого на проекта създадено с Photoshop)
  + serviceWorker.js
  + .gitignore
  + package-lock.json
  + package.json
  + README.md

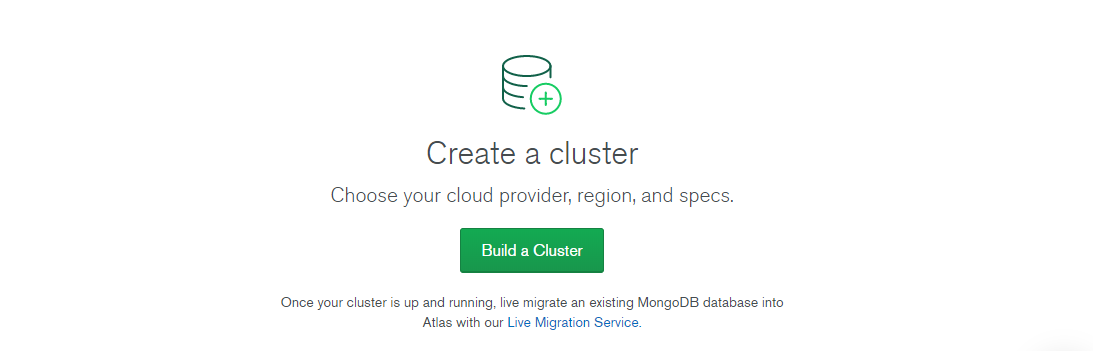
4. Описание на архитектурно значимите интерфейси

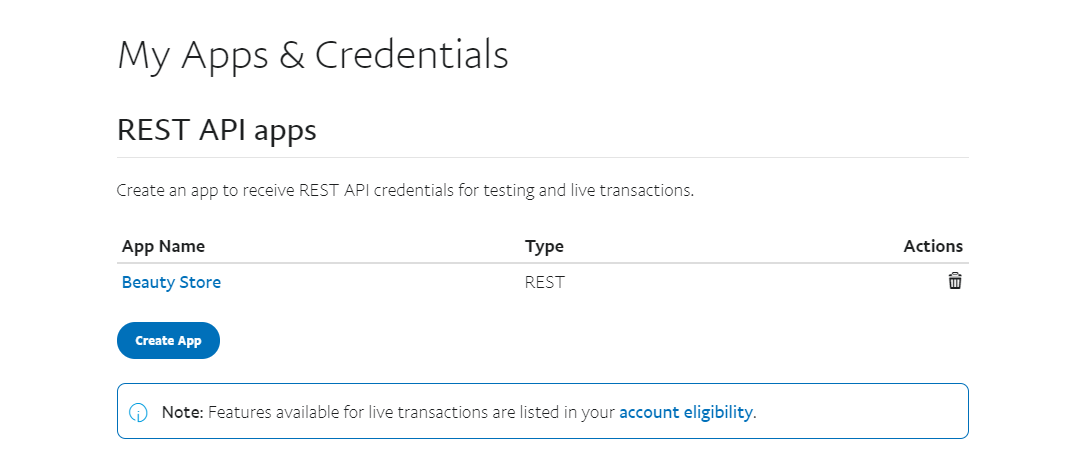
|  |  |  |
| --- | --- | --- |
| 1. API Resources (Node.js Backend) | | |
| **View name** | **Brief Descriptions** | **URI** |
| * 1. **Users** | GET *User Data* for all users, and POST new *User Data* (Id is auto-filled by *OBS* and modified entity is returned as result from POST request). Available only for *Administrators*. | */api/users* |
| * 1. **User** | GET, PUT, DELETE *User Data* for *User* with specified *userId*, according to restrictions decribed in UCs. | */api/users/{userId}* |
| * 1. **Login** | POST *User Credentials* (e-mail address and password) and receive a valid *Security Token* to use in subsequent API requests. | */api/login* |
| * 1. **Logout** | POST a logout request for ending the active session with *OBS,* and invalidating the issued *Security Token*. | */api/logout* |
| * 1. **Products** | GET *products* and POST new *products* (Id is auto-filled by *OBS* and modified entity is returned as result from POST request). | */api/products* |
| * 1. **Product** | GET, PUT, DELETE product for *products* with specified *productId*. | */api/products/{productId}* |
| * 1. **Orders** | GET orders and POST new *order* (Id is auto-filled by *OBS* and modified entity is returned as result from POST request). | */api/orders* |
| * 1. **Order** | GET, PUT, DELETE *order* for *orders* with specified *orderId*. | */api/orders/{ordersId}* |

5. Описание на начина на инсталиране и конфигуриране

2.1 Изтегляне и инсталиране на Node.js source-code или предварително построен инсталатор за вашата платформа.

2.2. [6] Изтегляне и инсталиране локално или безплатно ползване на MongoDB Atlas Global Database.

2.3. [7] Създаване на PayPal Developer Account и SANDBOX API CREDENTIALS за приложението.



6. [8] Потребителска документация за начина на работа със системата

npm run dev

Изпълнява nodemon server.js, което стартира сървъра.

npm start

Изпълнява приложението в development режим.

Отворете http: // localhost: 3000, за да го видите в браузъра.

Страницата ще се презареди, ако направите промени.

Грешките се визуализират в конзолата.

npm тест

Стартира test runner в интерактивен режим на гледане.

npm run build

Build-ва приложението за production в build папката.

npm run eject

Забележка: След тази операция не можете да се върнете!

Ако не сте доволни от build tool-а и избора на конфигурация, можете да го премахнете по всяко време.

Вместо това, той ще копира всички конфигурационни файлове и транзитивни dependency-та (Webpack, Babel, ESLint и т.н.) във вашия проект, така че да имате пълен контрол над тях. Всички команди, с изключение на eject, ще продължат да работят, но те ще сочат към копираните скриптове, за да ги настройвате.

7. Заключение - перспективи за бъдещо развитие

7.1 Добавяне на талони за отстъпка.

7.2. Възможност за проследяване на поръчката

7.3. Чат кутия

7.4. Категории продукти

7.5. Опции за продукт

8. Използвани материали

[1] Node.js (<https://en.wikipedia.org/wiki/Node.js>);

[2] Express (<https://en.wikipedia.org/wiki/Express.js>);

[3] React (<https://en.wikipedia.org/wiki/React_(JavaScript_library)>);

[4] Bootstrap (<https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)>);

[5] MongoDB (<https://en.wikipedia.org/wiki/MongoDB>);

[6] MongoDB Atlas (<https://www.mongodb.com/>);

[7] PayPal Developer (<https://developer.paypal.com/>);

[8] Available Scripts (<https://facebook.github.io/create-react-app/docs/available-scripts>);