**Софийски университет „Св. Климент Охридски“**

**Факултет по математика и информатика**



**Тема: AutoBot - уеб приложение за автомобилни обяви**

**Дисциплина: Разработка на клиент-сървър (fullstack) приложения**

**Автори: Ръководител:**

**/Ивайло Иванов 81187 3-ти курс/ /Траян Илиев/**

**/Димитър Калинков 44874 3-ти курс/**

**/Филип Иванов 44845 4-ти курс/**

Съдържание

[1. Описание 2](#_Toc492059167)

[2. Видове потребители 2](#_Toc492059168)

[3. Функционални и нефункционални изисквания 2](#_Toc492059169)

[3.1. Функционални изисквания 2](#_Toc492059170)

[3.2. Нефункционални изисквания 3](#_Toc492059171)

[4. Използвани технологии и библиотечни модули 3](#_Toc492059172)

[4.1. Клиентска част 3](#_Toc492059173)

[4.2. Сървърна част 3](#_Toc492059174)

[5. Взаимодействие между основните компоненти 4](#_Toc492059175)

[6. Основни изгледи 4](#_Toc492059176)

[7. Приложно програмен интерфейс на сървъра 5](#_Toc492059177)

[8. Инсталиране и конфигуриране на разработената система 6](#_Toc492059178)

[8.1. Конфигуриране на системата 6](#_Toc492059179)

[8.2. Конфигуриране на `mongodb` 6](#_Toc492059180)

[9. Потребителска документация за работа със системата 6](#_Toc492059181)

[10. Бъдещо развитие на системата 6](#_Toc492059182)

[11. Използвани материали 6](#_Toc492059183)

# 

# 

# Описание

AutoBot е приложение, което помага на хората да намерят търсеният от тях автомобил, като предоставя интуитивен начин за търсене на автомобил по множество характеристики и ценови диапазон. В допълнение на това позволява на регистрираните потребители да се абонират към конкретни изисквания и да получават известия при появата на нови обяви отговарящи на тези изисквания. Администраторите на приложението ще имат възможността да добавят нови производители, модели и видове екстри за търсене в портала.

# Видове потребители

Основните роли са:

* Анонимен потребител - това са потребители, които нямат регистрация и имат възможност само да разглеждат обяви без да се абонират за тях.
* Регистриран потребител (разширява анонимен потребител) - освен възможностите на анонимния потребител добавя възможност за абониране към обяви с дадени изисквания.
* Администратор – тези видове потребители могат да добавят нови опции за търсене и да управляват регистрациите на останалите потребители.

# Функционални и нефункционални изисквания

## Функционални изисквания

В таблицата са показани сценариите за употреба

|  |  |  |
| --- | --- | --- |
| **Име** | **Кратко описание** | **Потребители** |
| **Търсене на автомобил** | Позволява търсене на автомобили по различни критерии, като търсенето се извършва в системата, намират се резултатите и се сортират по най-близко съвпадение на търсените термини. Търси се по име и тагове на офертата. | Всички потребители |
| **Абониране за автомобили отговарящи на дадени изисквания** | Потребители могат да се абонират по различни критерии, на базата на който да получават нотификации, ако се появи нова оферта отговаряща на критериите им. | Регистрирани потребители |
| **Създаване на оферта** | Потребителите могат да създават оферта в сайта, като попълнят няколко основни полета за информация относно офертата. | Регистрирани потребители |
| **Изпращане на съобщения за оферта** | Потребителите могат да изпращат съобщения на потребителя създал дадена оферта, ако случайно се интересуват от нея. | Регистрирани потребители |
| **Управление на потребители** | Позволява на администратор да преглежда и променя потребителска информация, да създава и премахва потребители. | Администратори |
| **Управление на катерории** | Позволява добавяне на нови производители, модели и видове екстри за търсене в портала. | Администратори |

## Нефункционални изисквания

#да се направи

* Usability
* Security
* Performance

# Използвани технологии и библиотечни модули

## Клиентска част

Компонентите използвани за разработката на клиентската част на приложението са:

* [React](https://facebook.github.io/react/) - A javascript library for building user interfaces. Allows us to compose simple yet powerful UIs.
* [ReactRouter](https://www.npmjs.com/package/react-router) - Organizes the URL navigation in our React application.
* [Redux](http://redux.js.org/) - Predictable state container for JavaScript apps.
* [Webpack](https://webpack.js.org/) - Module bundler.

## Сървърна част

Компонентите използвани за разработката на сървърната част на приложението са:

* [Node](https://nodejs.org/) - an open-source, cross-platform JavaScript run-time environment for executing JavaScript code server-side.
* [Express](https://expressjs.com/) - Minimal and flexible Node.js web application framework.
* [Passport](http://passportjs.org/) - Simple, unobtrusive authentication for Node.js.
* [Mongoose](http://mongoosejs.com/) - Elegant mongodb object modeling for Node.js
* [Crypto](https://nodejs.org/api/crypto.html) - JavaScript implementations of standard and secure cryptographic algorithms.
* [Body-parser](https://www.npmjs.com/package/body-parser-json) - Node.js body parsing middleware
* [Cokie-parser](https://www.npmjs.com/package/cookieparser) - Fast cookie parser.
* [Eslint](http://eslint.org/) - The pluggable linting utility for JavaScript and JSX.
* [Babel](https://babeljs.io/) - A javascript compiler

# Взаимодействие между основните компоненти

#да се направи

* React component 1
* React component 2

# Основни изгледи

В таблицата са показани основните изгледи на приложението

|  |  |  |
| --- | --- | --- |
| **Име на изгледа** | **Кратко описание** | **URL** |
| **Home** | Предоставя възможност за избор на различните характеристики на автомобил и стартиране на търсенето. | / |
| **Login** | Позволява вписване в системата за регистрирани потребители | */login* |
| **Register** | Дава възможност за регистрация на нови потребители. | */register* |
| **Profile** | Визуализира текущите потребителски настройки и позволява промяната им. | */profile* |
| **Subscription** | Позволява създаване на различни условия срещу които да се проверяват новите обяви и да изпраща известия в портала и по електронна поща за всички удовлетворяващи потребителя оферти. Така че да може да се намери най-подходящия автомобил за потребителя. | */subscription* |
| **Users** | Позволява визуализиране на всички потребители. (Admin only) | */users* |
| **Manage user** | Дава възможност за промяна на настройките, ролите и паролата (reset), на произволен потребител (Admin only) | */users/{id}* |
| **Popular** | Визуализира най-търсените модели и марки автомобили. | */top* |
| **Category**  **Managment** | Позволява на администратирите да добавят нови производители, модели и видове екстри за търсене. | */autos* |

# Приложно програмен интерфейс на сървъра

За успешната създаване на приложението, са имплементирани следните крайни точки (endpoins) на сървъра:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Име на ресурса** | **Кратко описание** | **Метод** | **URI** | **Authentication** |
| **Auth** | Регистрира потребител в системата | POST | */register* | Anon |
| Автентикира потребител и го асоциира с куки | POST | */login* | *Anon* |
| Излиза от профила | GET | */logout* | *Login* |
| Предоставя сменя на паролата на потребителския профил. | POST | */passwd* | *Login* |
| **Profile** | Показва данните на потребителя. | GET | */profile* | *Login* |
| Позволява промяна на данните на потребителя. | POST | */profile* | *Login* |
| Предоставя информация за нотификациите на потребителя | GET | */profile/notifications* | *Login* |
| Маркира нотификация като прочетена | PUT | */pofile/notifications/:id* | *Login* |
| **Users** | Връща всички потребители на системата | GET | */users* | *Admin* |
| Регистрира потребител | POST | */users* | *Admin* |
| Връща информация за потребител спрямо подаден идентификатор | GET | */users/:id* | *Admin* |
| Заменя информацията за избран потребител с подадената | PUT | */users/:id* | *Admin* |
| Изтрива потребител спрямо подаден идентификатор | DELETE | */users/:id* | *Admin* |
| **Subscriptions** | Връща всички абонаменти на потребителя | GET | */subscriptions* | *Login* |
| Създава нов абонамент | POST | */subscriptions* | *Login* |
| Връща абонамент спрямо подаден идентификатор | GET | */subscriptions/:id* | *Login* |
| Изтрива абонамент спрямо подаден идентификатор | DELETE | */subscriptions/:id* | *Login* |
| **Messages** | Показва всички съобщения на потребителя | GET | */messages* | *Login* |
| Създава нова нишка за съобщения | POST | */messages* | *Login* |
| Връща всички съобщения от избрана нишка | GET | */messages/:id* | *Login* |
| Изпраща ново съобщение в избрана нишка | PUT | */messages/:id* | *Login* |
| Маркира избрана нишка като прочетена | PATCH | */messages/:id* | *Login* |
| **Offers** | Връща информация за всички оферти | GET | */articles* | *Anon* |
| Създава нов оферта в системата | POST | */articles* | *Login* |
| Връща информация за избрана оферта | GET | */articles/:id* | *Login* |
| Заменя информацията за избрана оферта с подадената | PUT | */articles/:id* | *Login* |
| Изтрива оферта от системата | DELETE | */articles/:id* | *Login* |
| Изпраща съобщение на собственика на офертата | POST | */articles/:id/offer* | *Login* |
| **Makers** | Връща информация за всички производители на автомобили | GET | */makers* | *Admin* |
| Създава нов производител на автомобили в системата | POST | */makers* | *Admin* |
| Връща информация за производител според подаден идентификатор | GET | */makers/:id* | *Admin* |
| Обновява информацията за производител спрямо подаден идентификатор | PUT | */makers/:id* | *Admin* |
| Изтрива информацията за производител спрямо подаден идентификатор | DELETE | */makers/:id* | *Admin* |
| **Models** | Добавя нов модел към производител на автомобили | POST | */makers/:id/models* | *Admin* |
| Връща информация за модел според подаден идентификатор | GET | */makers/:id/models/:id* | *Admin* |
| Изтрива модел според подаден идентификатор | DELETE | */makers/:id/models/:id* | *Admin* |
| **Search** | Търси оферти в системата спрямо името и таговете и връща резултати подредени по най-голяма близост до търсеното | POST | */search* | *Anon* |

# Инсталиране и конфигуриране на разработената система

## Конфигуриране на системата

1. Инсталирайте най-новите версии на `node.js` и `npm`
2. Клонирайте `git` хранилището - `git clone https://github.com/Del7a/vehicle-finder.git`
3. Инсталирайте зависимостите - `npm install`

## Конфигуриране на `mongodb`

1. Инсталирайте последната версия на `docker`
2. Изтеглете последната версия на `mongodb container` - `$ docker pull mongo`
3. Изпълнете командата - `$ docker run -p 27017:27017 --name autobot-mongo -d mongo`

# Потребителска документация за работа със системата

#да се направи #след създаването на потребителския интерфейс ще се добавят картинки описващи взаимодействието на потребителя с приложението

# Бъдещо развитие на системата

#да се направи #след като се уточни какво е текущото състояние и поведение на системата ще се добавят идеите за бъдещо развитие

# Използвани материали

<https://nodejs.org/en/docs/> (последно използвана на 06.2017)

<http://expressjs.com/en/4x/api.html> (последно използвана на 06.2017)

<https://facebook.github.io/react/docs/thinking-in-react.html> (последно използвана на 06.2017)