**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

[**Съдържание**](#_j9obhs7x06d1) **1**

[**Описание**](#_g9lqndknszn) **2**

[**Видове потребители**](#_p5z4ud6zn591) **2**

[**Функционални и нефункционални изисквания**](#_6uypjepz67ba) **2**

[Функционални изисквания](#_lf7dtaf2trn3) 2

[Нефункционални изисквания](#_p8spgbvett8p) 3

[**Използвани технологии и библиотечни модули**](#_k609k4xg4ea0) **3**

[Клиентска част](#_1y5myc6w86wl) 3

[Сървърна част](#_91ry4r1l59yq) 3

[**Взаимодействие между основните компоненти**](#_b15ul2gh6p7w) **4**

[**Основни изгледи**](#_y7bsxfpuhono) **4**

[**Приложно програмен интерфейс на сървъра**](#_5khtkvuabypl) **5**

[**Инсталиране и конфигуриране на разработената система**](#_hh2lpk1fjy44) **6**

[**Документация за начина на работа със системата**](#_vunpk1856vrv) **6**

[**Срещнати/преодолени трудности и перспективи за бъдещо развитие на системата.**](#_1dzznac17h1s) **6**

[**Използвани материали**](#_64i0otgxlicx) **6**

# 

# 

# 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

|  |  |  |
| --- | --- | --- |
| Сценарии на употреба | | |
| **Име** | **Кратко описание** | **Потребители** |
| **Търсене на автомобил** | Позволява търсене на автомобили по различни критерии, като извършва самото търсене из най-известните сайтове за подобни обяви и извежда систематизирана информация на нашия портал заедно с референция към източника на офертата. | Всички потребители |
| **Абониране за автомобили отговарящи на дадени изисквания** | *Регистрираните потребители могат да зададат критерии, по които искат да търсят автомобил и да се абонират за добавянето на нови обяви, удовлетворяващи изискванията им. През зададен интервал ще се изпращат като известия в портала и/или електронна поща под формата на систематизирана информация.* | *Регистрирани потребители* |
| **Популярни търсения** | Визуализира най-популярните производители, модели и критерии по-които останалите потребители на портала търсят из обявите. | *Всички потребители* |
| **Управление на потребители** | Позволява на администратор да преглежда и променя потребителска информация, да създава и премахва потребители. Да им променя ролите т.е. да създава нови администратори. Да изпраща email с възможност за избор на нова парола. | *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

* [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

* [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

|  |  |  |
| --- | --- | --- |
| **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

|  |  |  |
| --- | --- | --- |
| **Име на ресурса** | **Кратко описание** | **URI** |
| **Auth** | POST: Дава възможност на потребителя да се рагистрира в системата.  POST: Дава възможност на потребителя да влезе във вече съществуващия му профил.  GET: Излиза от профила  POST: Предоставя сманя на паролата на потребителския профил. | /register  /login  /logout  /passwd |
| **Profile** | GET: Показва данните на потребителя.  POST: Позволява промяна на данните на потребителя.  GET: Предоставя информация за абонаментите на потребителя. | */profile*  */profile*  */profile/subs* |
| **Makers** | GET: Позволява преглеждането на информацията за производителите на автомобили.  POST: Добавя нов производидел.  GET: Извежда информация за конкретен производител.  PUT: Позволява променяне на информацията за конкретен производител  DELETE: Изтрива производител на автомобили | */makers*  */makers*  */makers/:id*  */makers/:id*  */makers/:id* |
| **Models** | POST: Позволява добавянето на нов модел автомобил към произвежданите от даден производител.  GET: Извежда информация за моделите произвеждани от даден производител.  DELETE: Позволява изтриването на модел произвеждан от даден производител. | */makers/:if/models*  */makers/:id/models/:modelId*  */makers/:id/models/:modelId* |
| **Articles** | GET: Извежда информация за всички налични “оферти/обяви”.  POST: Добавя нова “оферта/обява”/  GET: Извежда конкретна обява.  PUT: Обновява информация за дадена обява.  DELETE: Изтрива обява с даден идентификатор. | */articles*  */articles*  */articles/:artId*  */articles/:artId*  */articles/:artId* |

# 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)