Технически университет – София

Факултет Компютърни и Софтуерни технологии

Дипломна Работа

Тема:

Мотористи БГ

Мобилно приложение целящо да обедини мотообщността.

Дипломант: Марин Николаев Маринов

Специалност: Компютърно и софтуерно инженерство

Научен ръководител: гл. ас Иван Станков

Съдържание

[1. Увод 3](#_Toc62031606)

[1.1 Цел и задача на дипломната работа 4](#_Toc62031607)

[1.2 Структура на дипломната работа 4](#_Toc62031608)

[2. Идеен проект 4](#_Toc62031609)

[3. Използвани технологии 5](#_Toc62031610)

[3.1 GitHub 5](#_Toc62031611)

[3.2 npm 5](#_Toc62031612)

[3.3 Node.js 6](#_Toc62031613)

[3.4 Cordova 6](#_Toc62031614)

[3.5 JavaScript 7](#_Toc62031615)

[3.6 CSS 7](#_Toc62031616)

[3.7 HTML 7](#_Toc62031617)

[3.8 jQuery 8](#_Toc62031618)

[3.9 Onsen UI 8](#_Toc62031619)

[3.10 Font awesome 8](#_Toc62031620)

[3.11 XML 8](#_Toc62031621)

[3.12 JSON 9](#_Toc62031622)

[3.13 Android SDK 9](#_Toc62031623)

[3.14 JDK 9](#_Toc62031624)

[3.15 Apache Ant 9](#_Toc62031625)

[3.16 Phonegap CLI 10](#_Toc62031626)

[3.17 Phonegap Desktop App 10](#_Toc62031627)

[3.18 Firebase 10](#_Toc62031628)

[3.18.1 Cloud Firestore 10](#_Toc62031629)

[3.18.2 Firebase Authentication 11](#_Toc62031630)

[3.18.3 Cloud Storage 11](#_Toc62031631)

[3.19 Atom IDE 12](#_Toc62031632)

[3.20 WinNMP 13](#_Toc62031633)

[3.21 Google Maps API 13](#_Toc62031634)

[3.22 Adobe Illustrator 14](#_Toc62031635)

[4. Архитектура 15](#_Toc62031636)

[4.1 Архитектура на приложението 15](#_Toc62031637)

[4.2 Архитектура на базата данни 17](#_Toc62031638)

[5. Разработка 20](#_Toc62031639)

[5.1 Инициализиране на проект 20](#_Toc62031640)

[5.2 Инсталиране на приставки за достъп до локация и камера 21](#_Toc62031641)

[5.3 Дизайн на приложението и основни библиотеки 22](#_Toc62031642)

[5.4 Структуриране на проекта 22](#_Toc62031643)

[6. Ръководство за потребителя 27](#_Toc62031644)

[7. Идеи за развитие 27](#_Toc62031645)

[8. Използвани източници 27](#_Toc62031646)

[9. Заключение 27](#_Toc62031647)

# Увод

Живеейки в XXI век, ние ставаме свидетели на изключително бързото развитие на технологиите. Те съпътстват нашия живот в абсолютно всеки аспект, без изключение, което ги прави и необходимост, и забавление. Всеки от нас притежава смартфон в наши дни и комуникацията и свързаността се развиват в много бързи темпове, Това от своя страна води до постоянно увеличаване в търсенето им и съответно до по-голям пазар. Използването на мобилни приложения за всякакви цели като игри, слушане на музика, споделяне на снимки, четене на книги, запознанства и комуникация с други. Това улеснява търсенето на подходящи уеб-сайтове за изброените цели, като ни предоставят готов продукт отговарящ на нуждите ни.

Писането на мобилно приложение за различните операционни системи изисква богати познания и умения за работа с определени езици за програмиране swift/objective-c за Apple и iOS базирани устройства, Java за андроид, C# за windows phone и др. Тук на помощ за разработчиците идват системите, които предлагат създаването на крос платформените приложения, които написани на една технология се превеждат до C#, swift, Java.

Статистиката показва, че потенциала на крос платформените приложения расте всяка изминала година. Най – високият процент на използването им е в Европа 38%. Двете големи предимства на крос-платформеното програмиране са: финансова ефективност и спестяващи време. Някои от най – добрите крос – платформени мобилни инструменти за разработка са: Xamarin, PhoneGap, Sencha, appcelerator, iFactr, kony, alphaSoftware, redhat, React Native, Vue Native.

## Цел и задача на дипломната работа

Целта на дипломната работа е да се създаде мобилно приложение, което да обедини мото общността на България. Да предлага следните функционалности:

* регистрация и вход
* създаване на профил
* споделяне на социални мрежи и данни за контакт
* организиране на чат стаи
* споделяне на снимки
* споделяне на постове
* споделяне на настоящо местоположение.
* запазване на потребители в „Любими“
* възможност за преглед на всички потребители

## Структура на дипломната работа

* Увод – изяснява целите и задачите на дипломната работа
* Идеен проект – представя основната идея на приложението
* Използвани технологии – дава информация какви технологии се използват за разработването на приложението
* Архитектура – дава информация за архитектурата и базата данни
* Разработка – разкрива начина по който е реализирано приложението
* Ръководство на потребителя – представя интерфейса и обяснява как се използва приложението
* Бъдещо развитие – дава информация за бъдещи планове и развитие на следващите версии на приложението
* Заключение – обобщава съдържанието и дава ясно дали целите за изпълнени успешно
* Използвана литература – съдържа всички материали, които са използвани за написването на дипломната работа

# Идеен проект

Основната идея на дипломната работа е да се разработи мобилно приложение „MOTORISTI BG“ , която основна цел е да предостави среда на определена група от хора с общи интереси, в която могат да си помагат, да споделят полезна информация и да си организират срещи. Основния интерфейс функционален, опростен, лесен за употреба и подходящ за всички мобилни устройства и таблети.

При регистрация всеки потребител може да качи своя снимка и сподели своите контакти като Facebook, Instagram, Email, телефонен номер, Скайп, както и външен сайт източник.  
Тази информация може да бъде открита в списъка с всички потребители или през картата.

Чрез картата потребителят може да види къде и кога са се намирали останалите потребители, като е отбелязано и неговото текущо местоположение. Да избира между сателитен изглед или обикновен, да обновява текущата си локация както и да използва street view изглед.

На социалната стена потребителя може да споделя текстови постове с или без снимка. Постовете могат да бъдат харесвани. Идеята е вдъхновена от Туитър, Стената на Фейсбук и Инстаграм.

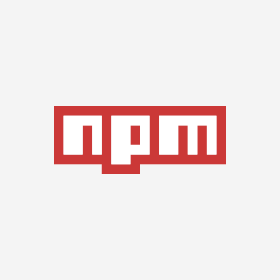
Чат групите са мястото където потребителите могат да си правят чат стаи на база интереси и да си пишат един с друг. Да качват снимка за корица на стаята и кратко описание. Идеята е вдъхновена от форумите на различните мото групи.

# Използвани технологии

## GitHub

GitHub е уеб услуга, използваща Git. Git е децентрализирана система за контрол на версиите на файлове. Създадена е от Линус Торвалдс за управление на разработката на Linux. Поради нуждата да се контролира огромната база от код на Linux ядрото, основна цел при разработката на Git е била бързината. Координатор на разработката на Git е Джунио Хамано. Всяка локална Git директория е хранилище с пълна история и възможности за следене на версиите. Това прави Git независим от мрежови връзки към централен сървър. Git е свободен софтуер и се разпространява под GPL лиценз версия 2.

## npm

npm (node package manager) е мениджър на пакети за езика за програмиране JavaScript. npm, Inc. е дъщерно дружество на GitHub, което осигурява хостинг за разработка на софтуер и контрол на версиите с използването на Git. npm е мениджърът на пакети по подразбиране за средата за изпълнение на JavaScript Node.js. Състои се от клиент на команден ред, наричан още npm, и онлайн база данни на публични и платени частни пакети, наречена npm регистър. Регистърът е достъпен чрез клиента, а наличните пакети могат да бъдат разглеждани и търсени чрез уеб сайта npm. Мениджърът на пакети и регистърът се управляват от npm, Inc.

## Node.js

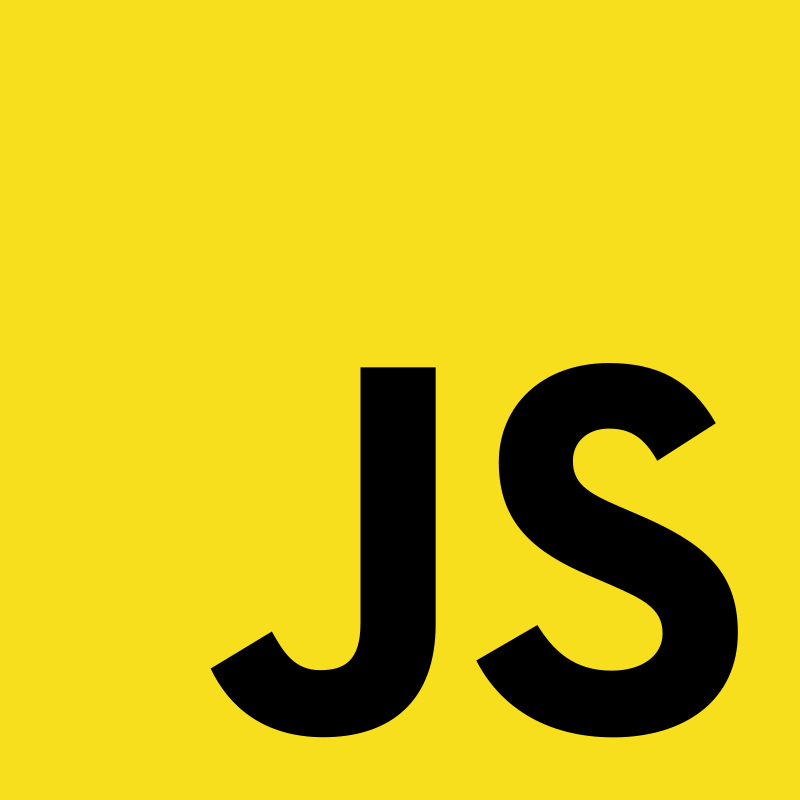
Node.js е многоплатформена среда за изпълнение на сървърни и мрежови приложения с отворен код. Приложенията се пишат на JavaScript и могат да се изпълняват в Node.js среда на Microsoft Windows, GNU/Linux, macOS и IBM.Node.js предоставя задвижвана от събития архитектура и неблокираща входно-изходна система за програмиране на приложенията (API), която оптимизира производителността и мащабируемостта на дадена програма. Тези технологии са често използвани за мрежови приложения, работещи в реално време. Node.js използва JavaScript двигателя V8 на Google, за да изпълнява код и голяма част от основните модули са написани на JavaScript. Node.js съдържа вградени библиотеки, които позволяват приложенията да работят като сървъри.

## Cordova

Apache Cordova (по-рано PhoneGap) е рамка за разработване на мобилни приложения, създадена от Nitobi. Adobe Systems закупува Nitobi през 2011 г., преименува го на PhoneGap и по-късно пусна версия с отворен код на софтуера, наречена Apache Cordova. Apache Cordova дава възможност на софтуерните програмисти да създават хибридни уеб приложения за мобилни устройства, използващи CSS3, HTML5 и JavaScript, вместо да разчитат на специфични за платформата API, като тези в Android, iOS или Windows Phone.

Apache Cordova се грижи да опакова в инсталационен файл в зависимост от операционната система (app.apk за Android, app.ipa за ios) CSS3, HTML и JavaScript. Получените приложения са хибридни, което означава, че те не са нито native мобилно приложение (защото цялото изобразяване на оформлението се извършва чрез уеб изгледи вместо вградената рамка на потребителския интерфейс на платформата), нито чисто уеб базирани (тъй като те не са само уеб приложения, а са пакетирани като приложения за разпространение (app.apk, app.ipa) и имат достъп до API на инсталираното устройство).

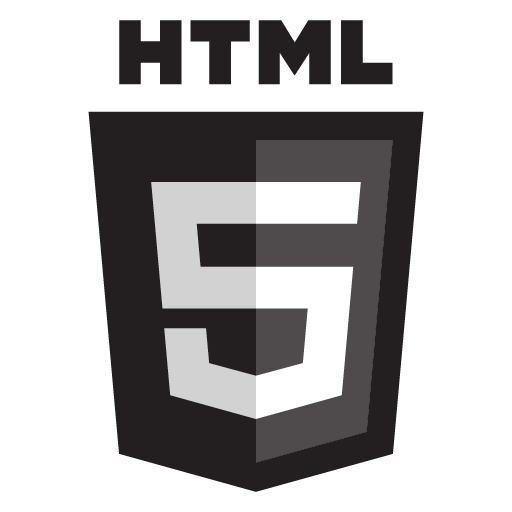
## JavaScript

Javascript (джаваскрипт) интерпретируем език за програмиране, разпространяван с повечето Уеб браузъри. Поддържа обектно-ориентиран и функционален стил на програмиране. Създаден е в Netscape през 1995 г. Най-често се прилага към HTML-а на Интернет страница с цел добавяне на функционалност и зареждане на данни. Може да се ползва също за писане на сървърни скриптове JSON, както и за много други приложения. JavaScript не трябва да се бърка с Java, съвпадението на имената е резултат от маркетингово решение на Netscape. Javascript е стандартизиран под името EcmaScript. JavaScript е разработен първоначално от Брендан Айк под името Mocha, като по-късно е преименуван на LiveScript и накрая на JavaScript. LiveScript е официалното име на езика когато за първи път бива пуснат в бета версиите на Netscape Navigator 2.0 през септември 1995 г., но е преименуван на JavaScript на 4 декември 1995г.

## CSS

CSS (Cascading Style Sheets) е език за описание на стилове (език за стилови листове, style sheet language) – използва се основно за описване на представянето на документ, написан на език за маркиране. Най-често се използва заедно с HTML, но може да се приложи върху произволен XML документ. CSS е създаден с цел да бъдат разделени съдържанието и структурата на уеб страниците отделно от тяхното визуално представяне. Преди стандартите за CSS, установени от W3C през 1995 г., съдържанието на сайтовете и стила на техния дизайн са писани в една и съща HTML страницата. В резултат на това HTML кода се превръща в сложен и нечетлив, а всяка промяна в проекта на даден сайт изисквала корекцията да бъде нанасяна в целия сайт страница по страница. Използвайки CSS, настройките за форматиране могат да бъдат поставени в един-единствен файл, и тогава промяната ще бъде отразена едновременно на всички страници, които използват този CSS файл.

## HTML

HTML е основният маркиращ език за описание и дизайн на уеб страници. HTML е стандарт в Интернет, а правилата се определят от международния консорциум W3C. Текущата версия на стандарта е HTML 5.0 В повечето случаи HTML кодът е написан в текстови файлове и се хоства на сървъри, свързани към Интернет. Тези файлове съдържат текстово съдържание с маркери – инструкции за браузъра за това как да се показва текстът. Например <маркер> Някакъв текст. </край на маркера>. Предназначението на уеб браузърите е да могат да прочетат HTML документите и да ги превърнат в уеб страници. Браузърите не показват HTML таговете, а ги използват, за да интерпретират съдържанието на страницата.

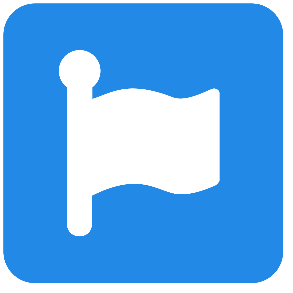
## jQuery

jQuery е разпространена библиотека на JavaScript, публикувана в началото на 2006 от Джон Резиг. В основата си jQuery опростява достъпа до всеки елемент на дадена уеб-страница, като по този начин позволява лесно изграждане на динамична функционалност в страниците. jQuery е безплатен и open source софтуер, лицензиран под MIT лиценз. jQuery се използва в 72% от 10000-те най-посещавани сайтове, което я прави най-популярната JavaScript алтернатива днес.

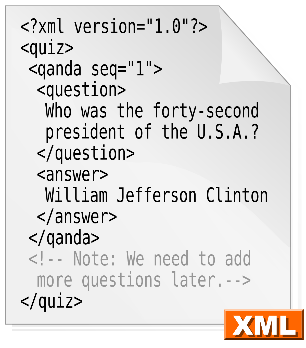
## Onsen UI

Onsen UI е библиотека с отворен код, която предоставя HTML компоненти подходящи за разработка на мобилни приложения със Cordova. Създадена през 2013г., първоначално е базирана на Angular JS, с поддръжка за jQuery. С излизането на версия 2, Onsen UI става самостоятелна JavaScript библиотека т.е. може да работи не зависимо без допълнителни библиотеки.

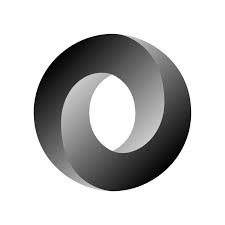
## 3.10 Font awesome

Преведено от английски-Font Awesome е набор от инструменти за шрифтове и икони, базиран на CSS и Less. Той е направен от Дейв Ганди за използване с Bootstrap, а по-късно е включен в BootstrapCDN. Font Awesome има 38% пазарен дял сред тези уебсайтове, които използват скриптове на шрифтове на трети страни на своята платформа, класирайки го на второ място след Google Fonts.

## 3.11 XML

XML е стандарт, дефиниращ правила за създаване на специализирани маркиращи езици, както и синтаксисът, на който тези езици трябва да се подчиняват. Сам по себе си той е безполезен, защото указва само как да бъде структуриран един документ, но не и какво означават отделните маркери. Ето защо е по-правилно да се нарече метаезик отколкото език – обикновените езици имат семантика, т.е. предават някаква информация, докато XML указва само граматиката (по-точно синтаксиса) на езиците, базирани на него.

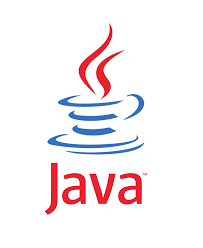
## 3.12 JSON

JSON, или JavaScript Object Notation, е текстово базиран отворен стандарт създаден за човешки четим обмен на данни. Произлиза от скриптовия език JavaScript, за да представя прости структури от данни и асоциативни масиви, наречени обекти. Въпреки своята връзка с JavaScript, това е езиково независима спецификация, с анализатори, които могат да преобразуват много други езици в JSON.Форматът на JSON първоначално е бил създаден от Дъглас Крокфорд (Douglas Crockford) и е описан в RFC 4627. Официалният Интернет медия тип за JSON е application/json. Разширението на файловете написани на JSON е .json.Форматът на JSON често е използван за сериализация и предаване на структурирани данни през Интернет връзка. Използва се главно, за да предаде данни между сървър и Интернет приложение, изпълнявайки функциите на алтернатива на XML.

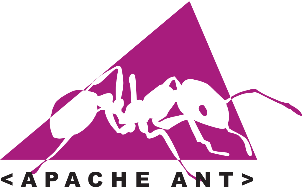
## 3.13 Android SDK

Android SDK е съвкупност от инструменти, с помощта на които се разработват приложения за операционна система Андроид. То включва в себе си Android Studio и официалната от Гугъл интегрирана работна среда (IDE) за операционна система Android. Представена от Гугъл през октомври 2009г. Език за разработка Java.

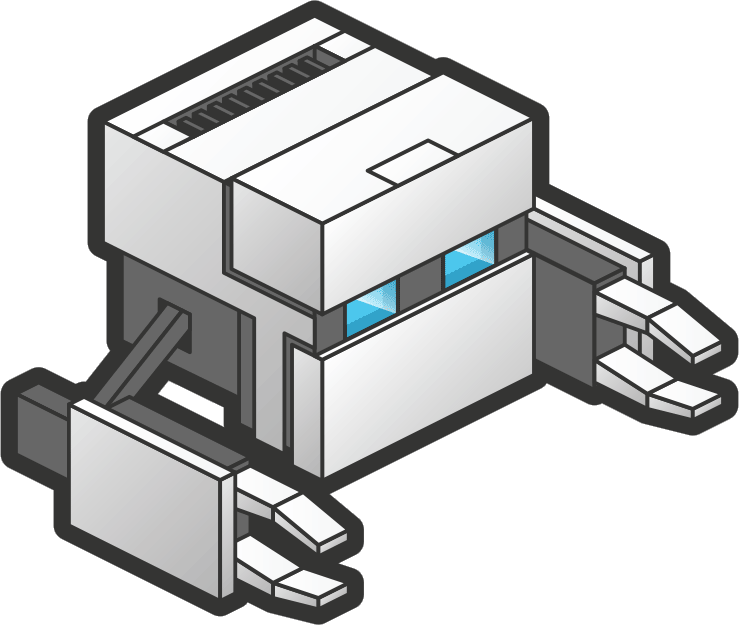
## 3.14 JDK

Java Development Kit (JDK) е реализация на една от платформите Java, Standard Edition, Java Platform, Enterprise Edition или Java Platform, Micro Edition, издадени от Oracle Corporation под формата на двоичен продукт, насочен към pазработчици на Java на Solaris, Linux, macOS или Windows. JDK включва частен JVM и няколко други ресурси, за да завърши разработването на Java приложение. От представянето на платформата Java тя е най-широко използваният комплект за разработка на софтуер

## 3.15 Apache Ant

Apache Ant е софтуерен инструмент за автоматизиране на процесите на изграждане на софтуер, произхождащи от проекта Apache Tomcat в началото на 2000 г. като заместител на Make make tool на Unix. Той е подобен на Make, но се реализира с помощта на езика на Java и изисква платформата Java. За разлика от Make, който използва формата Makefile, Ant използва XML, за да опише процеса на изграждане на кода и неговите зависимости. Издаден под лиценз Apache от Apache Software Foundation, Ant е проект с отворен код. Първа версия 19 юли 2000г.

## 3.16 Phonegap CLI

Comand Line Interface на PhoneGap предоставя интерфейс на командния ред за създаване на приложения на PhoneGap като алтернатива на използването на приложението PhoneGap Desktop. CLI на PhoneGap беше основният инструмент, използван за създаване на приложения на PhoneGap преди PhoneGap Desktop и съдържа допълнителни функции над PhoneGap Desktop за изграждане, стартиране и опаковане на приложения на PhoneGap за множество платформи.

## 3.17 Phonegap Desktop App

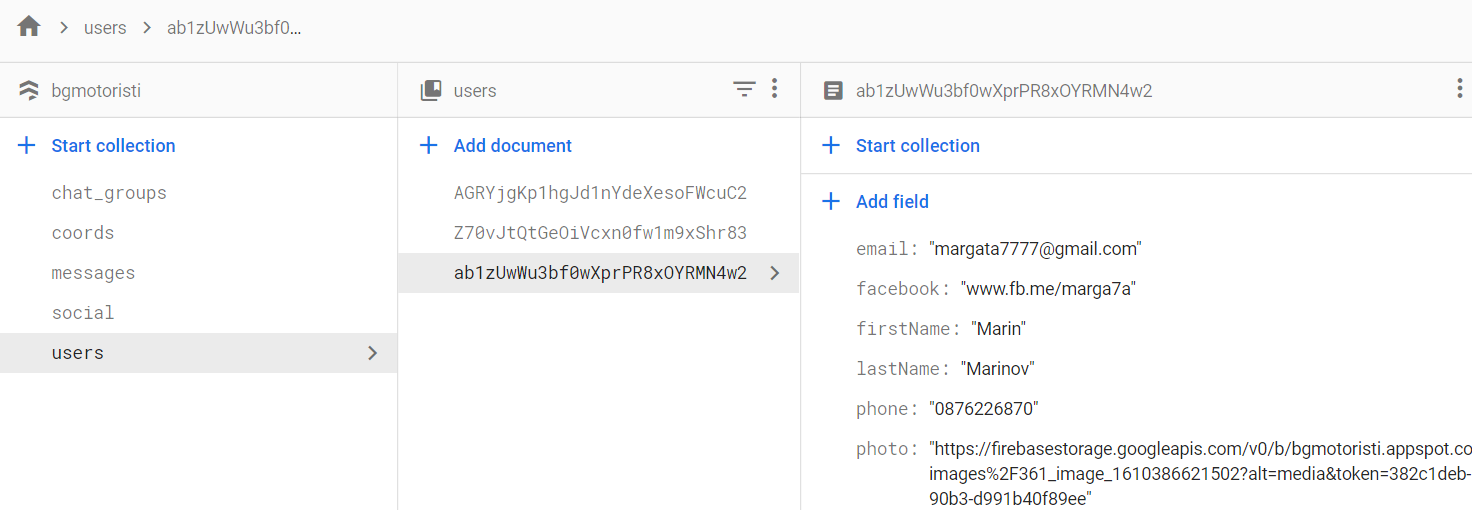
Настолното приложение PhoneGap е най-лесният начин да започнете с PhoneGap. Това е алтернатива на PhoneGap CLI, но използва същите задкулисни библиотеки, без сложна инсталация. Нямате нужда от git, npm или дори Node.js, просто го инсталирате на вашия работен плот, създавате проект и го стартирате на вашето устройство.

## 3.18 Firebase

Firebase е платформа, разработена от Google за създаване на мобилни и уеб приложения. Първоначално тя беше независима компания, основана през 2011 г. През 2014 г. Google придоби платформата и сега е водещото им предложение за разработка на приложения. Платформата Firebase предлага 18 услуги разделени в 3 основни групи:  
Development, Quality and Grow

### 3.18.1 Cloud Firestore

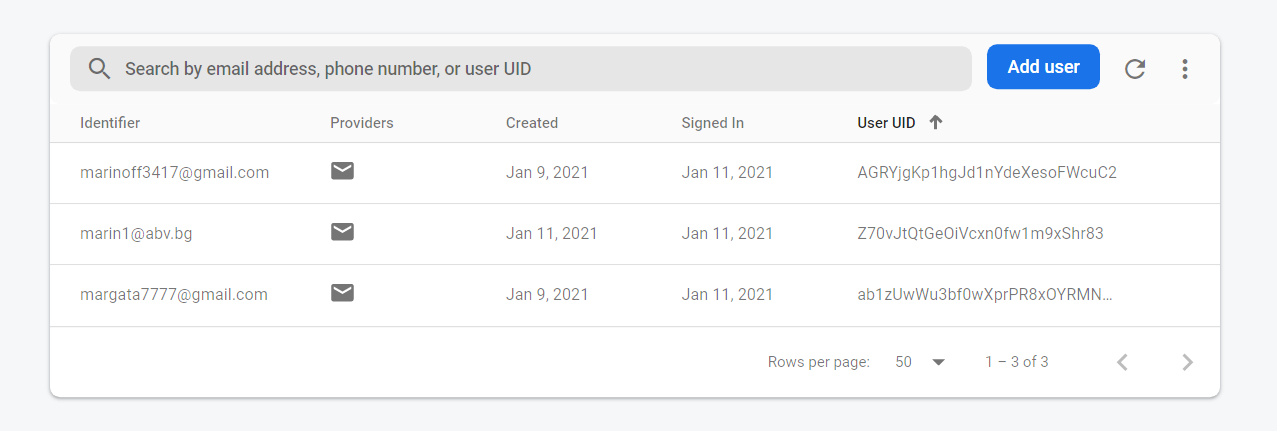
Cloud Firestore е нерелационна база данни подходяща за мобилни и уеб приложения. Предлага лесен запис на данни, синхронизация и извличане на данните във формат JSON. Лесно структуриране на данните с колекции и документи. Извличане на нужните данни чрез изразителни заявки. Всички заявки се мащабират с цел намаляване на размера от данни. Предоставя възможност да се закачим за избрана колекция и да ни информира при настъпване на промени, така приложението се синхронизира с базата в реално време и показва винаги актуална информация.



фигура 1. Представя структурата на базата данни вляво колекциите  
в средата документа за всеки запис и вдясно полетата на документа.

### 3.18.2 Firebase Authentication

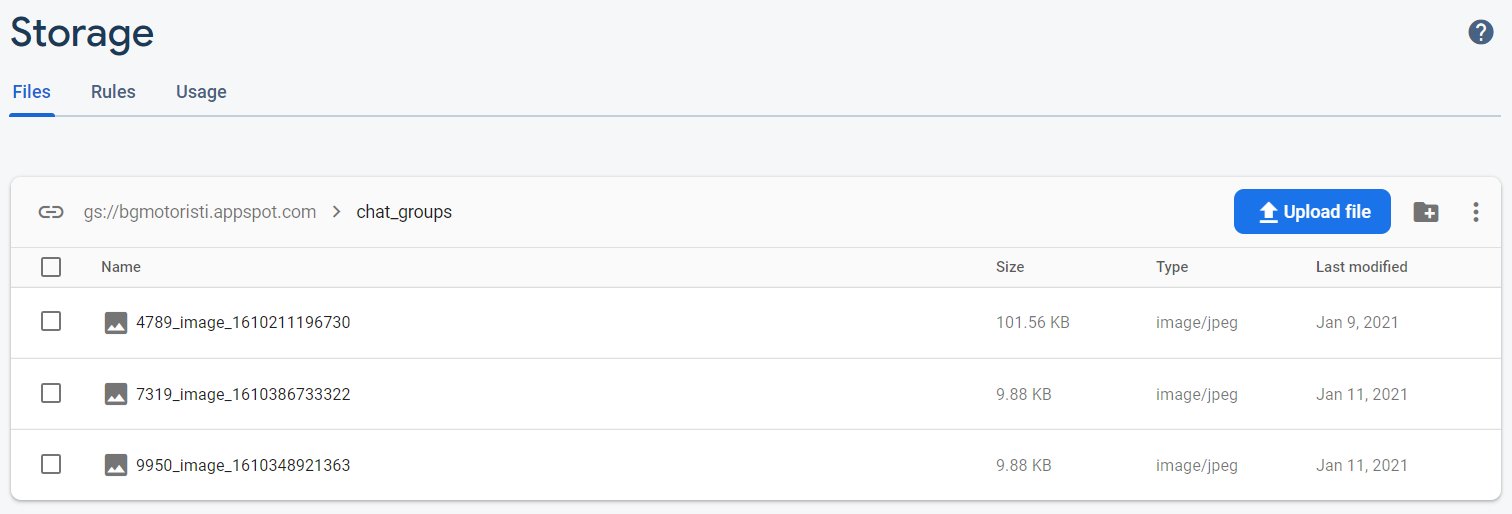
Firebase Authentication има за цел да улесни изграждането на защитени системи за удостоверяване, като същевременно подобри практиката за влизане и включване за крайни потребители. Той предоставя цялостно решение за самоличност, поддържащо акаунти с имейл и парола, удостоверяване на телефона и влизане чрез Google, Twitter, Facebook и GitHub и др. Създадена от същия екип, разработил Google Sign-in, Smart Lock и Chrome Password Manager, Firebase security прилага целия опит на Google за управление на една от най-големите бази данни за акаунти в света.



фигура 2. Списък с регистрираните потребители.

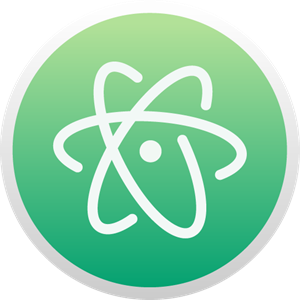
### 3.18.3 Cloud Storage

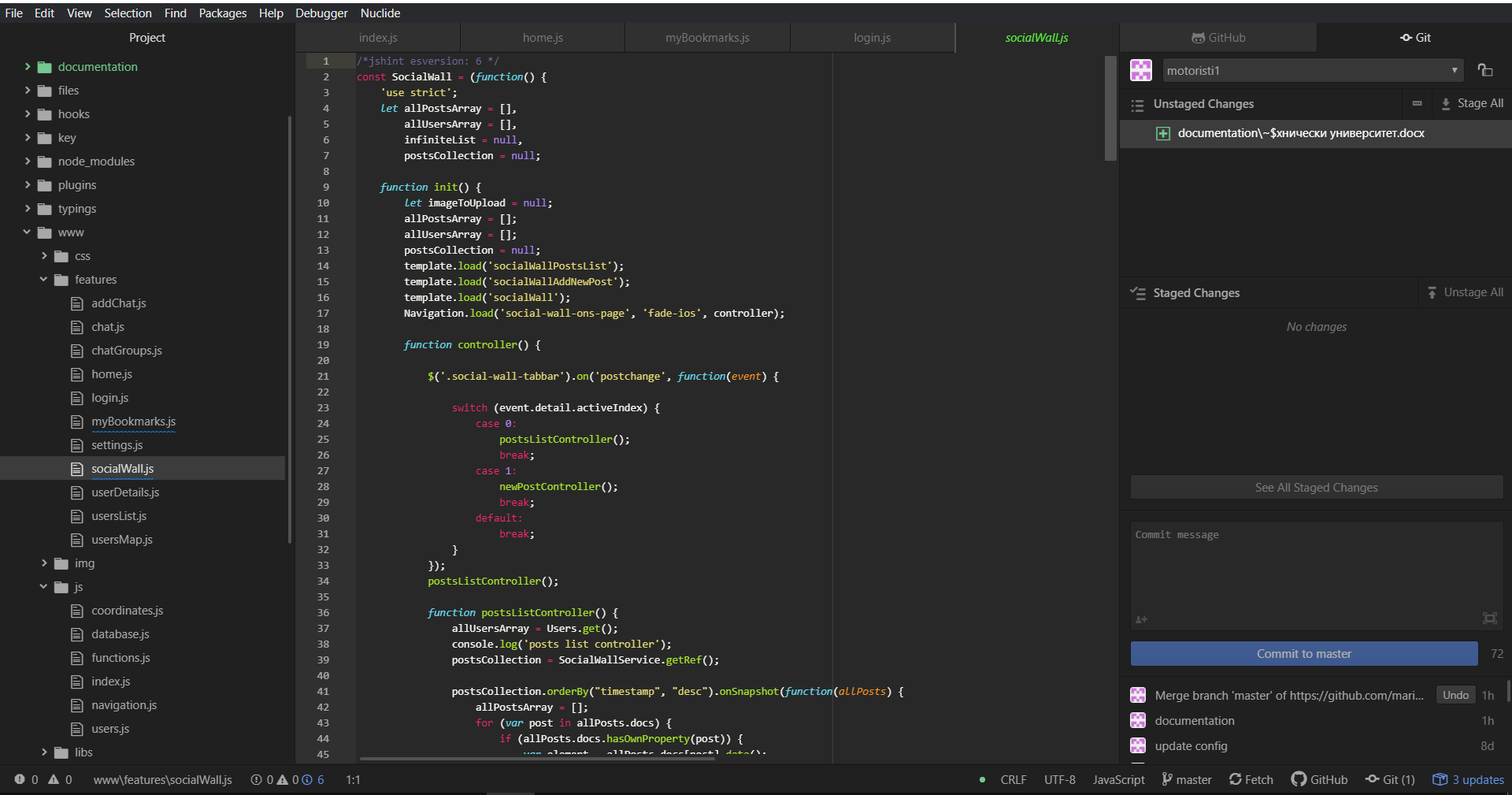
Cloud Storage е създадена за да може лесно и бързо да се съхранява снимки и видеа, като предоставя и добра синхронизация между себе си и Cloud Firestore.



фигура 3. Списък с качените файлове в Cloud Storage.

## 3.19 Atom IDE

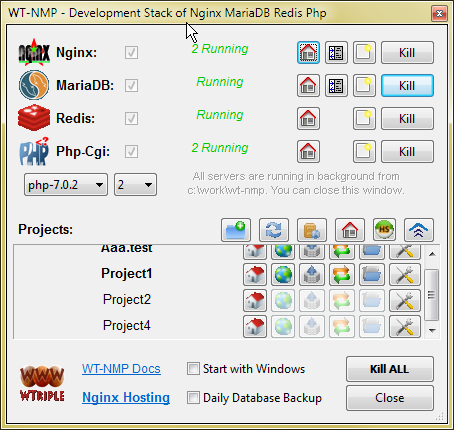
Atom е безплатен текстов редактор с отворен код и подходящ за macOS, Linux и Microsoft Windows с поддръжка за приставки, написани на JavaScript, и вграден Git Control, разработен от GitHub. Atom е настолно приложение, създадено с помощта на уеб технологии. Повечето разширения за с безплатен лиценз и се поддържат от общността. Atom е базиран на Electron (познат като Atom Shell), рамка (framework), която позволява разработката на крос платформени настолни приложения, използващи Chromium и Node.js. Написан е на Coffee Script и LESS. Първата версия излиза на 25 юни 2015г. Едно от неговите предимства е, че всеки може да си го персонализира, както пожелае използвайки HTML, CSS и JavaScript.



фигура 4. Изглед от Atom IDE

## 3.20 WinNMP

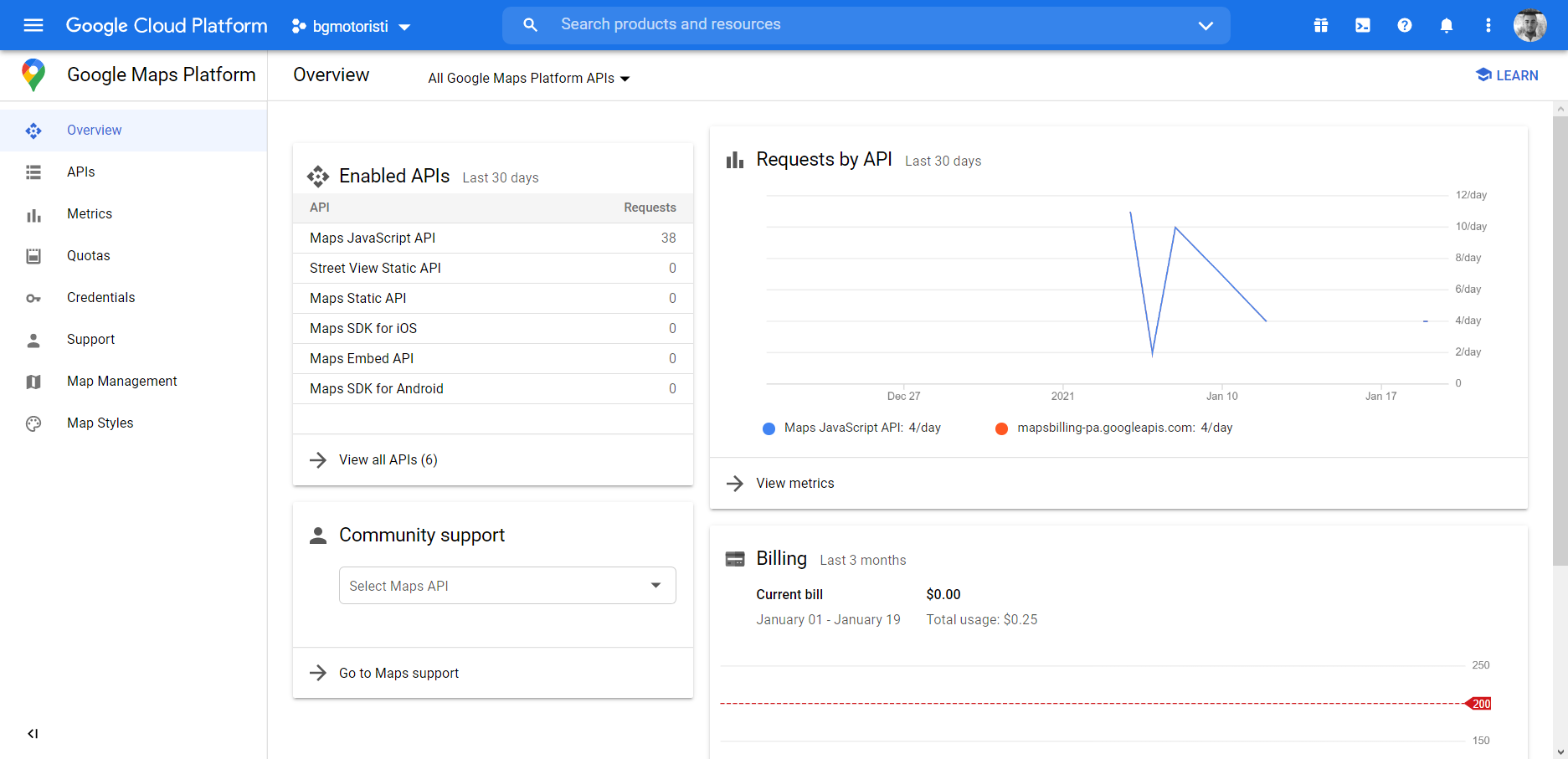
WinNMP - Nginx MariaDB MongoDB Redis Php 7 стек за разработка за Windows Лек, бърз и стабилен сървърен стек за разработване на php mysql приложения на Windows, базиран на отличния уеб сървър Nginx. По-лека алтернатива на XAMPP и WAMP, с Composer, Adminer, LetsEncrypt сертификати, WinSCP, wp-cli, множество PHP версии, проекти и виртуални сървъри.



фигура 5. WinNMP мениджъра работи и хоства нашето приложение

## 3.21 Google Maps API

Гугъл карти (на английски: Google Maps) е картографска услуга на Гугъл, с която могат да се разглеждат географски карти в произволен уеб браузър. Картите са с променлив мащаб и показват широк спектър от информация за разглеждания регион или местност. Има допълнителен режим за разглеждане на сателитни изображения с или без картографско съдържание, както и теренни карти с изолинии на височините над морското равнище. За територията на България се извършва автоматичен превод на български на населените места и улиците. Google Maps API позволява да се вграждат Гугъл карти в собствени уебстраници посредством JavaScript. Приложно-програмният интерфейс на Google Карти и Гугъл земя предоставя определен брой функции за работа с тези карти (така както е в www.maps.google.com и www.code.google.com/apis/earth/) позволяващо да се добави съдържание през различни услуги, и така да се създават пълноценни карти на произволен сайт. Настоящата версия на Maps API е 3. За да използвате гугъл карти трябва да си направите акаунт в Google Cloud platform и да дадете данни за разплащателна сметка, защото услугата не е безплатна.



фигура 6. Изглед от Google Cloud Platform

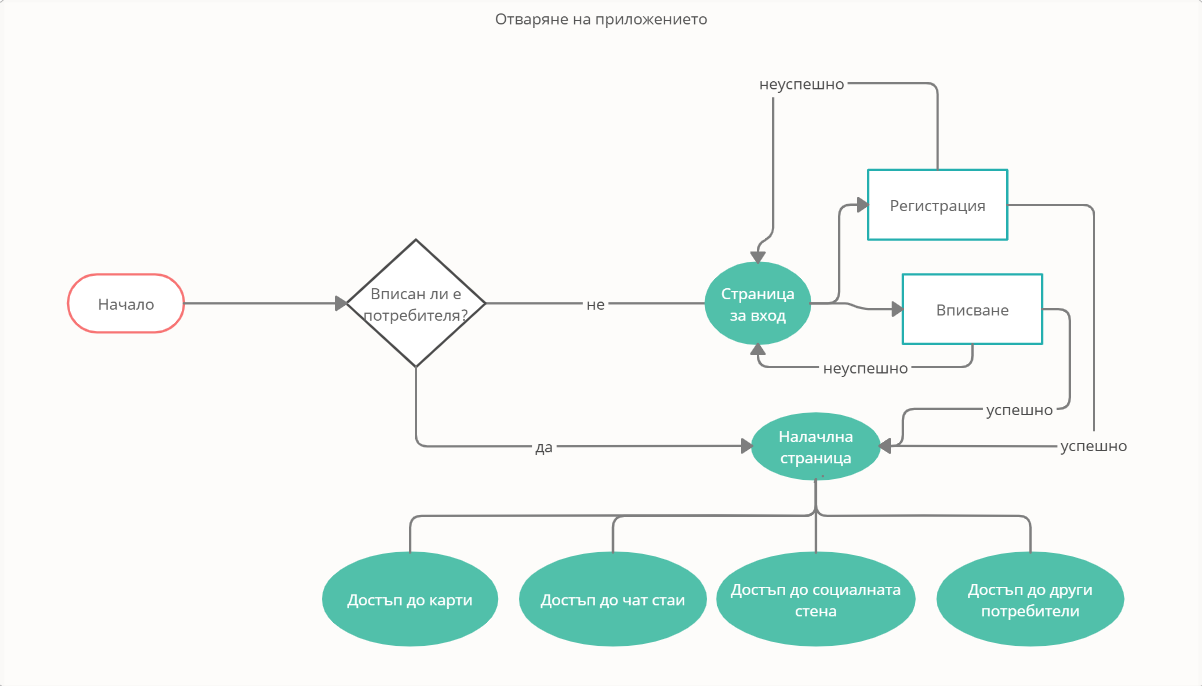
## 3.22 Adobe Illustrator

Adobe Illustrator е редактор и програма за дизайн на векторни графики, разработена и продавана от Adobe Inc. Първоначално проектирана за Apple Macintosh, разработването на Adobe Illustrator започва през 1985 г. Последната версия, Illustrator CC 2021, излезе на 20 октомври 2020 г. и е 25-то поколение в продуктовата линия. Adobe Illustrator е оценен като най-добрата програма за редактиране на векторни графики през 2018 г. от списание PC Magazine. Позволява да се създават всякакви графични изображения – от уеб и мобилни графики до лога, икони, илюстрации на книги, опаковки на продукти и билбордове. С възможност за използването на Illustrator на настолен компютър и iPad.

# Архитектура

## 4.1 Архитектура на приложението

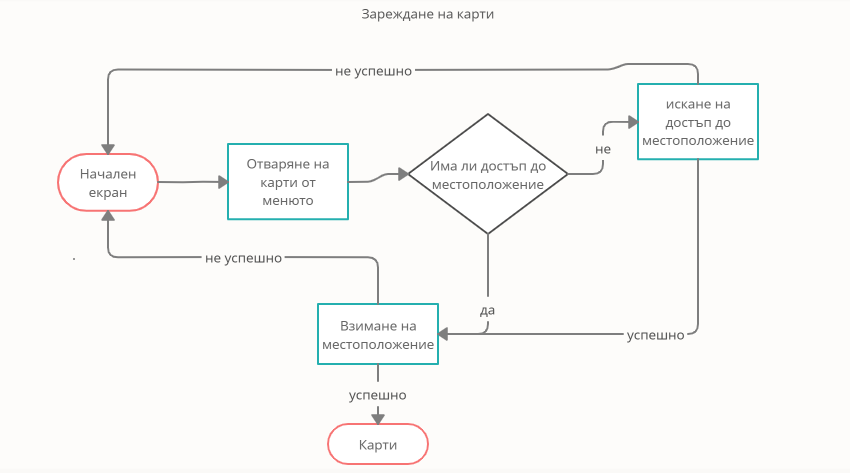
Тук ще разгледаме архитектурата на мобилното приложение и услугите които ще предоставя на потребителите



фигура 7. Блок-схема на зареждане на приложението

Всички потребители имат еднакви права и поради тази причина те могат да извършват едни и същи операции. Всички операции могат да се извършват само и единствено след вписване/оторизиране на потребителя. Без да се е вписал успешно, той няма достъп до нито един от екраните в приложението освен страницата на вписване.

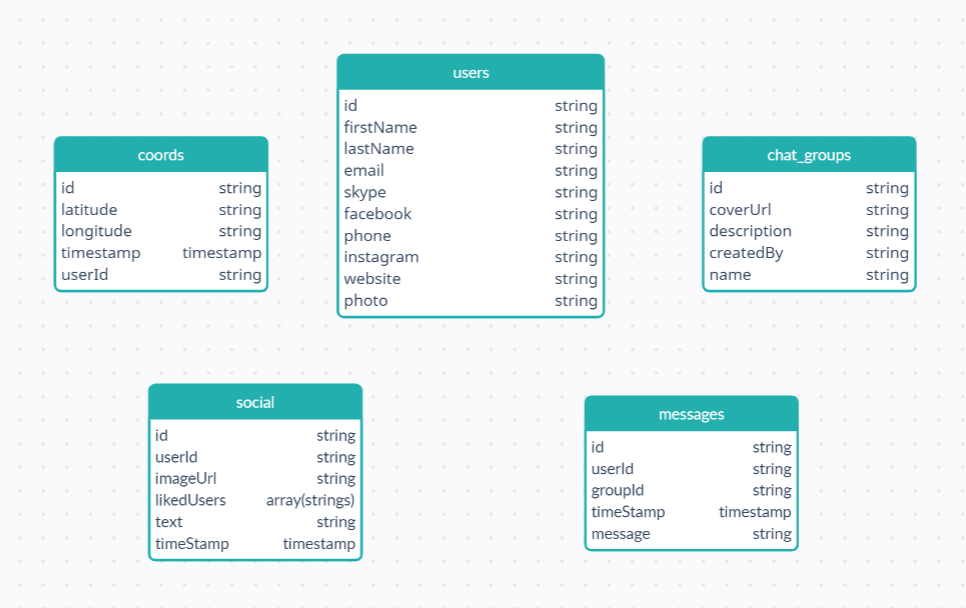
Когато потребител отвори приложението за първи път той ще види страницата за вход. На нея може да се впише с имейл и парола или да се регистрира с имейл, парола име и фамилия. При грешка той остава на този екран. При успешно вписване получава достъп до всички функции на апликацията и е изпратен на начална страница. Веднъж вписан в приложението, потребителя не е нужно да въвежда отново своите данни, а ще бъде пратен директно на начална страница. Тя е отправна точка към всички други страници.



фигура 8. Блок-схема на отваряне на карта с потребители

Блок схемата представя последователността при опит за отваряне на картата с потребители. Потребителя трябва да активира GPS на устройството си и да сподели своята локация. Ако условията не са изпълнени, той ще бъде помолен да стартира GPS и даде достъп на приложението до локацията. В противен случай той няма да има достъп до картите и ще остане на началната страница. Ако си е пуснал GPS и е дал достъп на приложението до локацията картите ще се отворят без да се изисква нищо от потребителя.

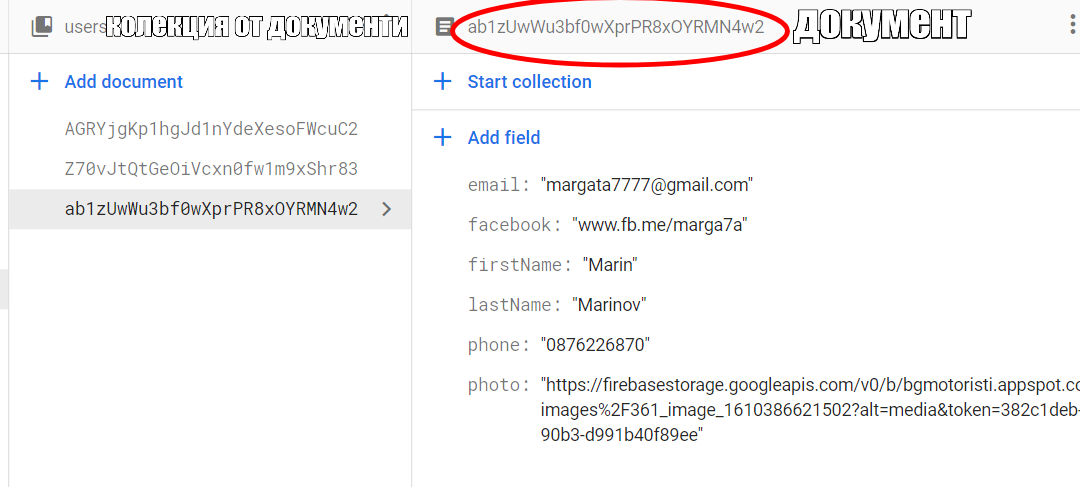
## 4.2 Архитектура на базата данни



фигура 9. Всички колекции и полетата за всеки документ

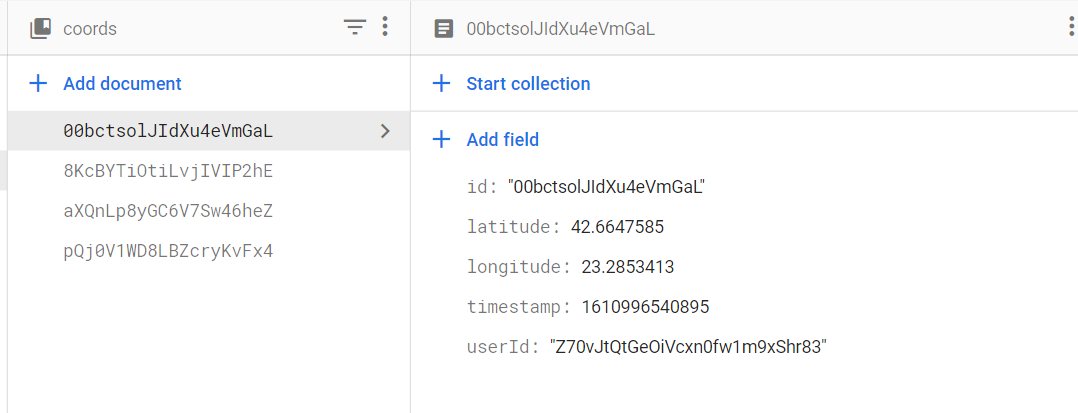
Базата данни е нерелационни и не съществуват връзки между отделните колекции.  
Идеята на този тип бази данни е информацията да се разбие на колкото може по малки парчета от данни (документи) като всеки документ държи в себе си информация за друг документ с който е свързан.Затова съм разбил всяка една страница от приложението в отделна колекция. Имаме една за потребителите (users) и 4 за картата, социалната стена, чат групите и съобщенията в тях, като общото между тях е, че всяка една има поле, което сочи към потребителя свързан с тях.

В “users” се пазят данни за всеки един потребител, неговия имейл(email), име(first\_name), фамилия(last\_name), както и ако е споделил социалните си мрежи и снимка (facebook, instagram,phone,website,skype, photoUrl).

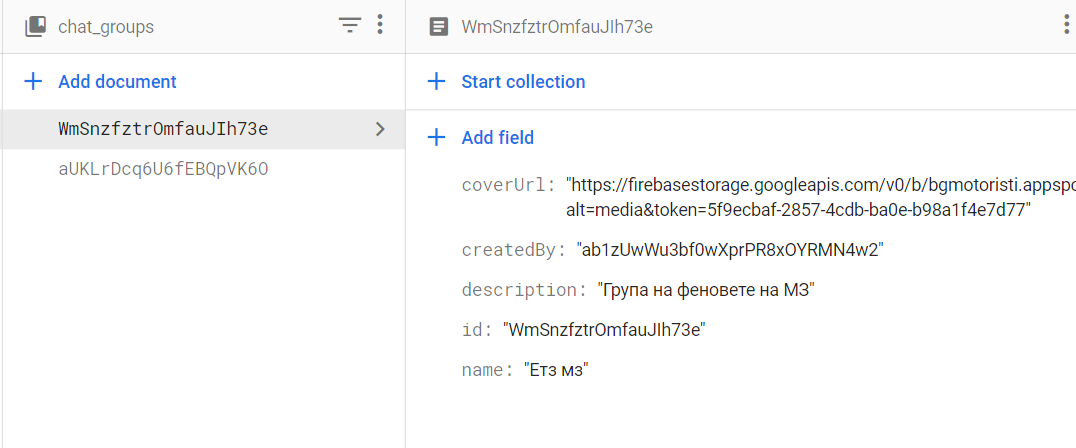


фигура 10. Колекция “users” и с три документа и полетата в един от тях

В coords -> id на документа , времето в което са взети координатите “timestamp”, самите координати ( lantitude ,longitude) както и “userId” - id-то на потребителя на чиито координати отговаря записа.

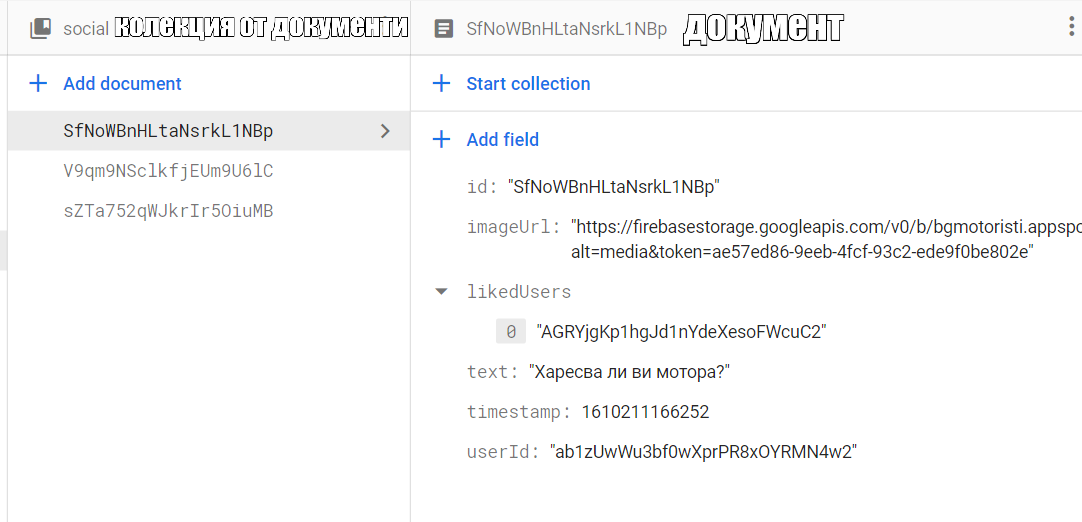


В chat\_groups имаме id на документа, корица на групата (coverUrl) , точно време кога е създадена (timestamp) и кратко описание за какво се отнася тя (description). Тук също имаме id на документа на потребителя (createdBy) създал чат стаята.



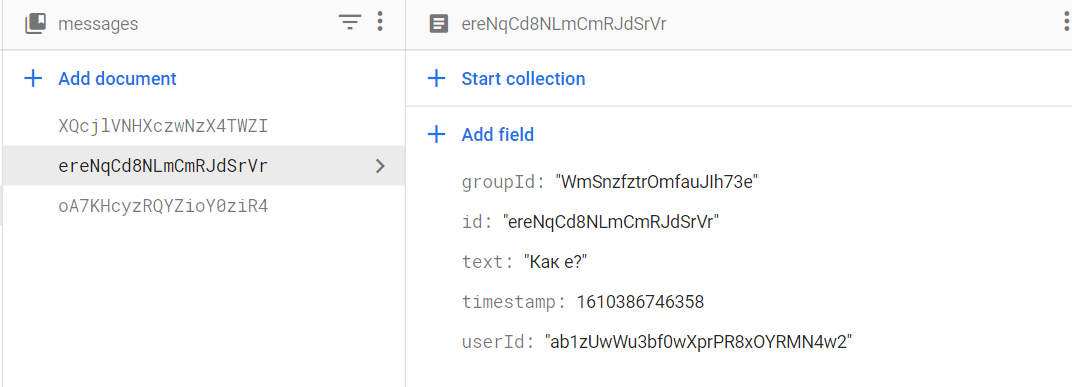
фигура 11. Колекция “chat\_groups” със два документа в себе си и полетата, които се съдържат в един документ.

В social ползваме “text” което е съдържанието на поста, “imageUrl” – снимката на поста, „timestamp“ – точно време кога е създаден поста, „likedUsers“ е масив от id-та на потребителите харесали поста и “userId” за потребителя създал поста.



фигура 12. Колекция “social” с три документа и полетата които се съдържат в един от документите.

В messages имаме следните полета: „id“ на съобщението, “userId” което показва ид-то на потребителя, който го е написал, “groupId” групата към която пренадлежи съобщението, “timestamp” – точно време кога е написано и “text” – съдържанието на съобщението



фигура 13. Колекция “messages” и документ на съобщение, който показва потребителя, групата на които принадлежи и неговото съдържание.

# Разработка

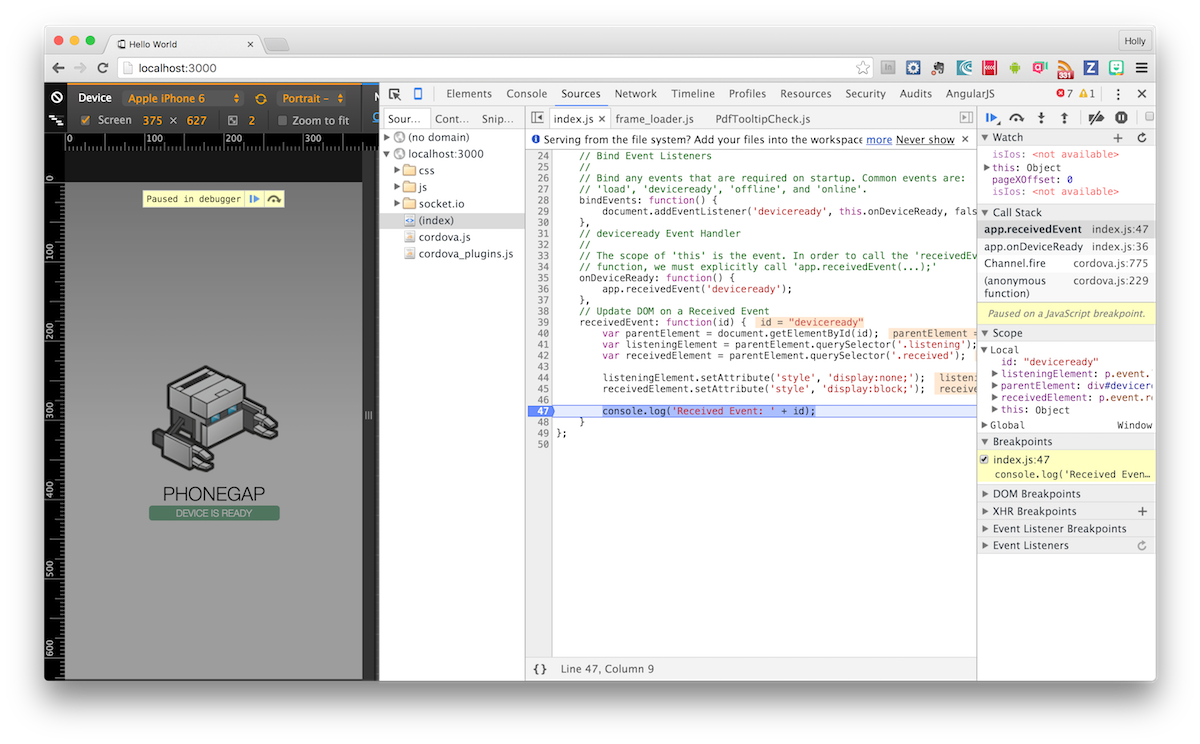
Основен език за писане на приложението е JavaScript, като е избран функционален стил пред обектно ориентиран. Всеки скрипт е вкаран в модул (клас), който представлява IIFE (Immediately invoked function expression), която скрива функциите и променливите вътре като създава вътрешен scope за всяка една функция на класа и не позволява на скриптовете да се презаписват или да си пречат. Така една функция или променлива от един клас, няма да презапише функция или променлива със същото име от друг клас. Направени така модулите споделят набор от методи към глобалния scope, но не може да се правят промени по тяхната функционалност от глобалния scope.

## Инициализиране на проект

За да стартиране проекта са ни нужни всички посочени технологии от точка 3.  
Започваме със създаването на начален проект чрез Phonegap CLI, отваряме терминал в желаната от нас директория („желателно е да е папка която се хоства от WinNMP“) и изпълняваме командата:



Това ще инициализира празен проект за нас със необходимите файлове за да може след това да пакетираме готовия продукт в инсталационен файл за съответната операционна система. След като успешно сме създали приложението стартираме WinNMP, отваряме браузъра и избираме директорията където се намира приложението.



фигура 14. Изглед от браузъра при стартиране на ново създаденото приложение.

## Инсталиране на приставки за достъп до локация и камера

За имаме достъп до камерата на устройството и неговото местоположение ни е нужно да инсталираме приставки. Тук използваме Cordova CLI, чрез следните команди изпълнени в директорията на проекта добавяме нужните ни приставки.

cordova plugin add cordova-plugin-camera

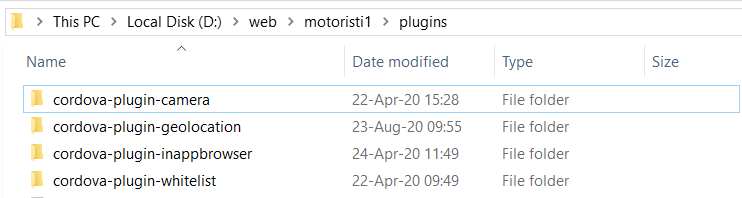
cordova plugin add cordova-plugin-geolocation

Нужни са ни още две приставки едната е задължнителна.

cordova plugin add cordova-plugin-whitelist

А другата за да може да отваряме страниците към социалните мрежи на потребителите без да затваряме приложението

cordova plugin add cordova-plugin-inappbrowser

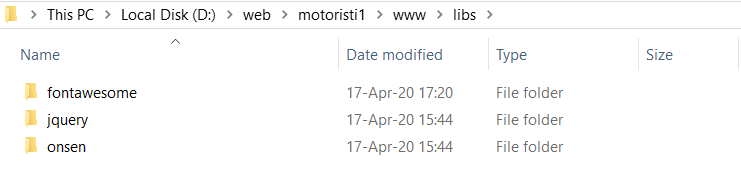


фигура 15. изглед от папка “plugins” след като сме добавили всички приставки успешно

## Дизайн на приложението и основни библиотеки

Нужни са ни само още няколко неща преди да започнем да наливаме основната логика в приложението. За дизайна използваме Onsen UI. Той е подходящ за екрани на телефони и таблети. В него има заложена функционалност за Single Page Application, т.е. да сменяме страниците в приложението без да се презареждат, която ще доразвием, за да работи в контекста на приложението. След Onsen UI, нашето приложение има нужда от иконки за тях използваме Fontawesome Icons. А за обработка на събития, анимации и работа с DOM дървото залагаме на jQuery.

Така добавяме нашите библиотеки в папка www (в нея се съдържат всички файлове, които не са част от Phonegap и после ще се пакетират. Нашите скриптове и стилове, библиотеки, иконки).



фигура 16. Изглед от папка “libs” след като сме добавили всички нужни библиотеки успешно

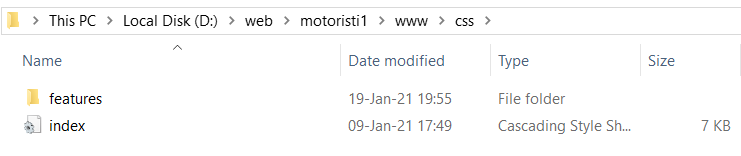
## Структуриране на проекта

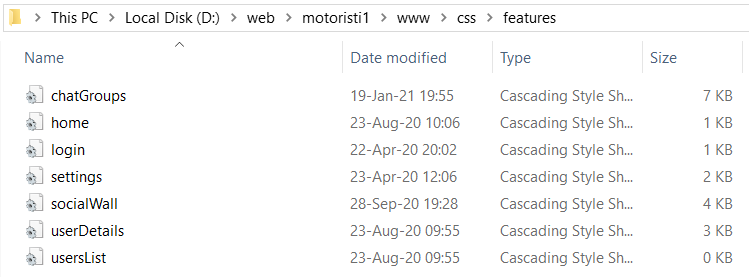
След като имаме основните библиотеки трябва да подготвим структурата на проекта. В нашия случай папките в папка “www” са разделени по следният начин:

www/css/ - в нея се съдържат стиловете на приложението, които сме написали

www/css/index.css – един файл, който държи глобалните стилове валидни за всички страници

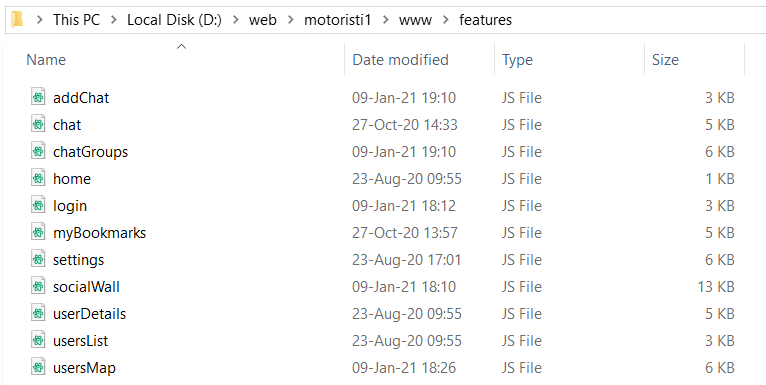
www/css/features/ - папка в която файловете са разделени по името на страницата, за която се отнасят например: www/css/features/home.css – тук стиловете са валидни само за началната страница.





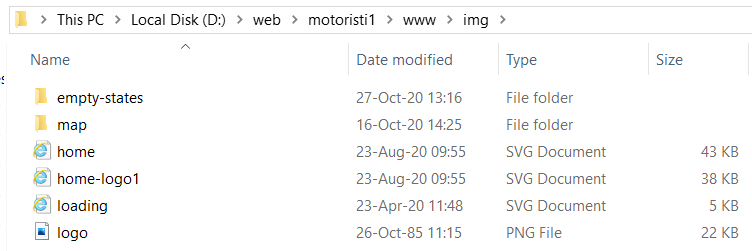
фигура 17 и 18. Структура на css файловете в приложението

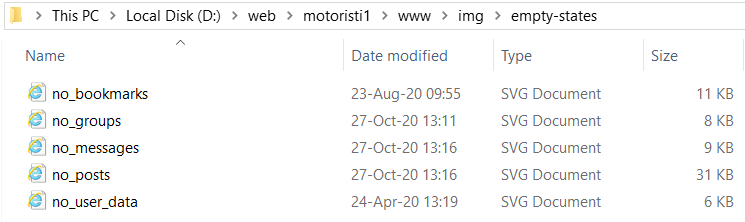
www/features/ - папка в която файловете са разделени по името на страницата. Тук всеки скрипт се отнася само за съответната страница например: www/features/home.js – този скриптов файл се отнася само за началната страница.

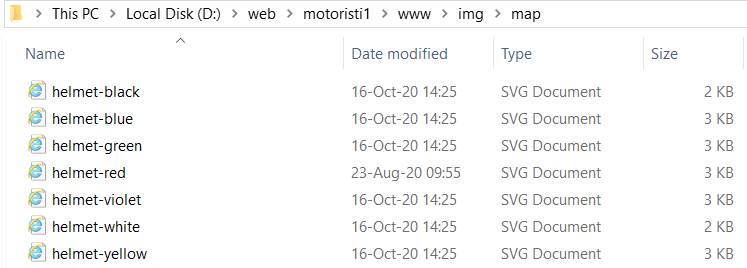


фигура 19. Всяка една страница в приложението си има скриптов файл

www/img – тук се намират всички графични изображения, които са част от апликацията.







фигура 20,21,22. показват как са разделени графичните елементи, които се използват в апликацията

www/js – тук са основните скриптове на приложението, те не са свързани само с един изглед

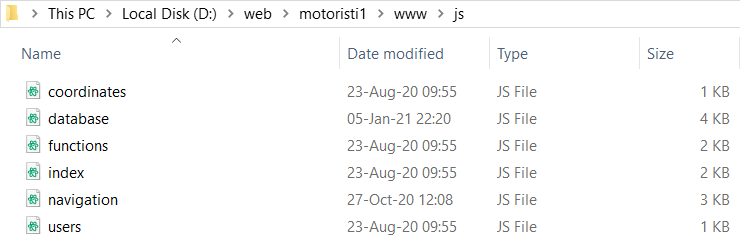
www/js/index.js отправна точка на апликацията от тук тя се стартира и инициализира базата данни

www/js/database.js – основен файл в нашето приложение, грижещ се за връзката между базата данни и приложението, дали е вписан потребителя или не (на база това апликацията ще поеме в различна посока към начална страница или страницата за вход) и синхронизацията на данните като потребителите и координатите на които се намират.

www/js/navigation.js – Файл грижещ се за навигирането в приложението, какво трябва да се случи преди да се смени екран, след като се смени, как да бъде заредена страницата (каква анимация да използва при смяната на екрана).

www/js/coords.js и www/js/user.js това са два файла с капсулована информация за координатите на всички потребители и информация за самите тях, тъй като тази информация се използва на различни екрани е изнесена тук при основните скриптове.

www/js/functions.js – споделени функции между всички файлове като валидиране на линк и конвертиране на изображения.



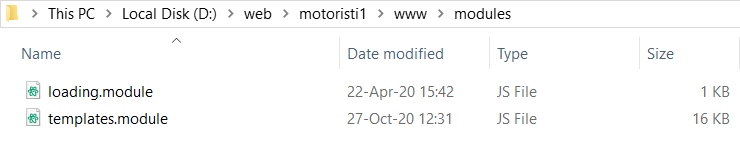
фигура 23. Основните скриптове нужни на приложението

www/libs/ - тук се намират библиотеките, които описахме в точка 5.3

www/modules/ - тук се намират скриптове, които манипулират екраните по някакъв начин:

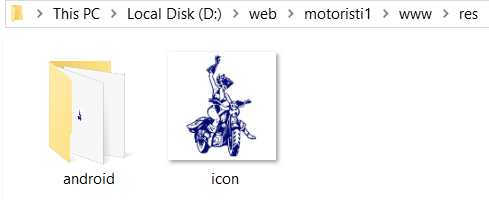
www/modules/loading.module.js – Той се грижи докато се зарежда приложението или тегли някакви данни, записва данни – да не може потребителя да използва другите функционалности.

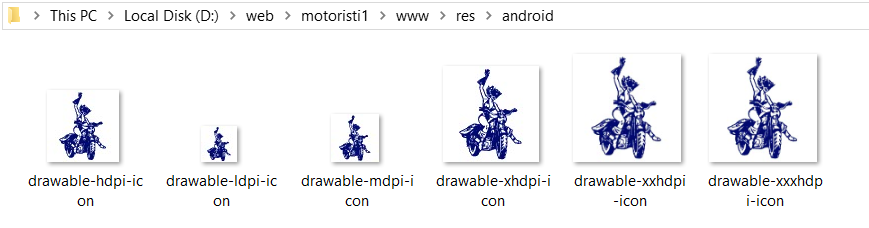
www/modules/templates.module.js – Той държи HTML темплейтите за всеки един екран и се грижи преди отварянето на всеки един от тях да го зареди в DOM дървото, но не и преди това, за да пести от ресурси.



фигура 24. Модулите които манипулират страниците на апликацията

www/res – Тук се намира основната иконка на приложението в размер 1024x1024, както и нарязаните иконки за андроид по спецификация с Adobe Illustrator.





фигура 25 и 26. Иконките нужни за приложението в съответния размер

* drawable-ldpi-icon.png размери 36х36
* drawable-mdpi-icon.png размери 48х48
* drawable-hdpi-icon.png размери 72х72
* drawable-xhdpi-icon размери 96x96
* drawable-xxhdpi-icon размери 144х144
* drawable-xxxhdpi-icon размери 192х192

www/services – Това са скриптове, които обработват по някакъв начин информацията, но не влияят пряко върху страниците и не могат да ги манипулират.

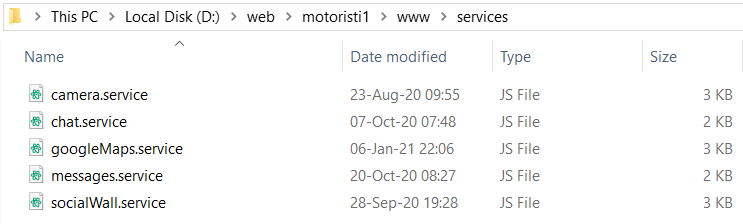
www/services/camera.service.js – Споделя методи за отваряне на камерата за снимка или отварянето на галерията, както и записването на снимката в буферната памет за обработка

www/services/chat.service.js – Споделя методи за осъществяване на постоянна връзка в реално време между приложението и чат стаите, както и запис на нова група в базата данни

www/services/googleMaps.service.js – Споделя метод за взимане на текущото местоположение на потребителя

www/services/messages.service.js - Споделя методи за осъществяване на постоянна връзка в реално време между приложението и съобщенията в избрана чат стая, както и запис на ново съобщение в базата данни

www/services.socialWall.service.js – Споделя методи за осъществяване на постоянна връзка в реално време между приложението и постовете в социалната стена , както и запис на нов пост в базата данни, харесване или не харесване на пост.



фигура 27. Всички services в приложението

# Ръководство за потребителя

# Идеи за развитие

# Използвани източници

# Заключение