**Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования**

**«КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

ИНСТИТУТ ФИЗИКИ

КАФЕДРА РАДИОАСТРОНОМИИ

Направление: 03.03.03. — «Радиофизика»

КУРСОВАЯ РАБОТА

**Heatmap by WebGazer**

**Работа завершена:**

"\_\_\_"\_\_\_\_\_\_\_\_2020 г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(A.A Ямолдин)

**Работа допущена к защите:**

Научный руководитель

ассистент

"\_\_\_"\_\_\_\_\_\_\_\_\_\_\_2020 г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(К.?. Скобельцин)

Заведующий кафедрой

канд. физ.-мат. наук, доцент

"\_\_\_"\_\_\_\_\_\_\_\_\_\_\_2020 г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(А.Д. Акчурин)

Казань — 2020

**СОДЕРЖАНИЕ**

Введение……………………………………………………………………….….3

ГЛАВА 1. Свободный браузерный айтрекер WebGazer.js……………………..……………………4

1.1. Особенности айтрекера WebGazer.js……………………..……..........…..4

1.2. Достоинства и недостатки айтрекера WebGazer.js……..........……...…10

ГЛАВА 2. Свободная библиотека hetmap.js…………………………………………..…………………………….13

2.1 Особенности библиотеки heatmap.js

2.2 Достоинтсва и недостатки heatmap.js

ГЛАВА 3.ОСНОВНЫЕ СВЕДЕНИЯ О СОЗДАНИИ ОДНОСТРАНИЧНЫХ WEB-ПРИЛОЖЕНИЙ С ПОМОЩЬЮ ФРЕЙМВОРКА ANGULARJS…………………………………………..…………………………….13

3.1. AngularJS и совместимые языки программирования…………………….13

3.2. Среда разработки WebStorm IDE……………………………………….….14

3.3. Структура AngularJS приложений в среде WebStorm IDE для фреймворка AngularJS……………………………...15

ГЛАВА 4. ПРАКТИЧЕСКАЯ ЧАСТЬ…………………………..….……

3.1. Подготовка к работе приложения в index.html ………………21

3.2. Создание AngularJS приложения

3.2. Запуск айтрекера WebGazer.js……………………………………...……………22

3.3. Обмен данными между скриптами внутри web-приложения …………………………………………………26

3.3 Создание тепловой карты с помощью heatmap.js

3.4 Деплой приложения на хостинг GitHub Pages

Заключение…………………………………………………………….……..…..30

Список использованных источников………………………………….…….….31

Приложение………………………………………………………………………32

**ВВЕДЕНИЕ**

При разработке современных web-приложений часто требуется сбор данных о действиях пользователя в режиме реального времени. В настоящий момент для сбора данных существует несколько готовых решений, такие как JavaScript-библиотека TensorFlow(https://www.tensorflow.org/js/), JavaScript-библиотека WebGazer.js(https://webgazer.cs.brown.edu/#home). При работе таких скриптов часто требуется передача цифровых данных с различных датчиков, либо приборов на персональный компьютер. Это можно сделать по различным проводным и беспроводным цифровым интерфейсам, например Ethernet, Wi-fi, Bluetooth, USB. Каждый из них требует отдельного подхода и специализированного оборудования. Актуальным является применение интерфейса Ethernet в связи с его высокой стабильностью и скоростью передачи данных.

Поэтому перед нами была поставлена цель осуществить передачу данных на компьютер с помощью созданного нами AngularJS web-приложения, вебкамеры, дальнейшей их обработки с помощью свободной библиотеки WebGazer.js и наглядное представление этих данных в виде тепловой карты построенной посредством JavaScript-срипта heatmap.js. Для достижения поставленной цели потребовалось решить следующие задачи:

* ознакомление со свободной библиотекой WebGazer.js;
* ознакомление со свободным скриптом heatmap.js
* изучение языка программирования JavaScript;
* изучения языка разметки HTML;
* Изучение языка таблиц стилей CSS
* Изучение фреймфорка AngularJS
* Работа с git и деплой приложения на хостинг GitHub Pages
* написание программы для внедрения библиотеки WebGazer.js в AngularJS web-приложение, а также для передачи данных в библиотеку отображения heatmap.js для посроения тепловой карты движения глаз по web-странице.

**ГЛАВА 1. Свободный браузерный айтрекер WebGazer.js**

**1.1.** **Особенности айтрекера WebGazer.js**

WebGazer.js - это свободно-предоставляемая JavaScript-библиотека для отслеживания местоположения взгяда пользователя на web-странице в режиме реального времени, использующая данные, полученные с помощью обычных веб-камер.

Webgazer.js разработана на основе исследования, проведеного Университетом Брауна[WebGazer: Scalable Webcam Eye Tracking Using User Interactions]. Работа над файлом калибровки была разработана в контекте курсового проекта с целью улучшения обратной связи WebGazer.js, предложенного доктором Джеральдом Веббером и его командой: доктором Клеменсом Зейдлером и Кай-Чунг Леуном.

Модель отслеживания взгяла, используемая в этой библиотеке, самокалибруется путём наблюдения за взаимодействиями веб-пользователя с веб-страницей и оттачивает зависимости между особенностями положения глаз пользователя и его местоположением на мониторе.

WebGazer.js - библиотека, написанная полностью на JavaScript и с помощью всего нескольких строк кода может быть интегрирована в лбьой веб-сайт. Webgazer работает полностью на клиентской стороне браузера, что гарантирует безопасность и сохранность персональных данных, так как никая информация не сможет быть отправлена на сервер без согласия пользователя на предоставление прав доступа к пользовательской веб-камере.

**1.2.** **Достоинства и недостатки айтрекера WebGazer.js**

В данной работе была выбрана именно эта библиотека в силу общедоступности программного обеспечения и средств разработки. Кроме того, WebGazer.js делает простой работу с пользоваельскими данными для разработчика, но, в отличие от иных систем, даѐт ряд превосходств для преподавателей, студентов и радиолюбителей:

* **Легкость в использовании.** Самое большое преимущество

библиотеки Webgazer.js - это готовая к использованию структура.Всё, что требуется для работы библиотеки это влючение нескольких строк кода в шапку HTML страницы.

* + **Широкая браузерная пожержка**.Еще одно большое преимуществоWebGazer.js-это возможность работы в большинстве современных браузеров за счет использования getUserMedia/Stream API[https://developer.mozilla.org/en-US/docs/Web/API/MediaDevices/getUserMedia] для доступа к веб-камере. Этот метод поддерживается большинством современных браузеров[https://developer.mozilla.org/en-US/docs/Web/API/MediaDevices/getUserMedia#Browser\_compatibility]
    - **Легкий доступ к выходным данным библиотеки.** Для получения предсказанных данных местоположения пользовательского взгляда достаточно добавить несколько строк JavaScript кода в тело HTML страницы.
    - **Программное обеспечение с открытым исходным кодом.**  WebGazer.js это библиотека с открытым исходным кодом, что дает большое преимущество для разработчка в плане понимания механизмов предсказания местоположения взгяда пользователя, реализованных в программе посредством просмотра “внутренностей” библиотеки.
  + **Нет необходимости в специальном программном обеспечении**. Большинство веб-камер поддерживаются данной библиотекой.

Но у WebGazer также есть и недостатки:

* **Структура.** СтруктураWebGazer.jsтакже является её недостатком.Присоздании проекта требуется совершать межскриптовые и межфункциональные переходы, а громоздкость библиотеки не позволяет осуществлять это быстро, а главное, зачастую теряется логическая связь между функциями, переменными и областями видимости.
* **Малочисленное сообщество**.В связи с малой извесностью данной библиотеки в сети практически нет примеров использования WebGazer.js в веб-разработке, в связи с этим во многих аспектах программы приходилось быть пионером.
* **Скудность документации.** Официальная документация описывает лишь малочисленные аспекты работы данной библиотеки.
* **Большой вес.** Объем данной библиотеки составляет 2.4МБ, что значительно тормозит загрузку страницы на стороне пользователя.
* **Возвращенные координаты относительно окна просмотра.** Предсказанные значения возвращаются относительно окна просмотра и не учитывают вертикальную и горизонтальную прокрутку страницы, что заставляет нас вносить собственную корректировку на прокрутку.

**ГЛАВА 2. Свободная библиотека heatmap.js**

**2.1. Особенности библиотеки heatmap.js**

heatmap.js - это свободно предоставляемая JavaScript- библиотека для визуализации трехмерных данных на веб-странице в режиме реального времени.

heatmap.js получает на вход трехмерные данные, две размерности координат точки (x, y) и третья размерность вес данной точки. В нашем случае все точки равноправны, поэтому имеют одинаковый вес.

Принцип построения тепловой карты состоит в следующем: если точка с координатами (x,y) и весом (value) в пределах какой-то области (radius) попадает в область уже существующей точки, то вес данной области будет равен сумме весов этих точек. Суммарное значение делится на максимальное, в резуальтате того, какую часть данная сумма занимает от максимального значения, выбирается каким цветом отображать данную область.

heatmap.js - это библиотека, написанная полностью на JavaScript и с помощью всего нескольких строк кода может быть интегрирована в любой веб-сайт. heatmap.js работает полностью на клиентской стороне браузера, что гарантирует безопасность и сохранность персональных данных.

**2.2. Достоинства и недостатки heatmap.js**

В данной работе была выбрана именно эта библиотека в силу общедоступности программного обеспечения, а такде имеет ряд достоинств для преподавателей, студентов и радиолюбителей:

* **Бытрота.** Во время последнего обновления библиотеки был добавлен

новый модуль обновления данных, в результате чего имеется возможность построения тепловой карты в режиме реального времени.

* **Мощность.** Бибиотека позволяет обрабатывать до 40 000 точек.
* **Легкость в использовании.** Для получения возможнсти использовать

данную библиотеку достаточно включить несколько строк кода в шапку HTML страницы.

* **Легкий вес.** Объем данной библиотеки всего 25кБ, в результате чего

браузер тратит минимальное время на загрузку страницы.

Но у heatmap.js также есть и недостаток:

* **Отсутсвие нормировки на динамический максимум.** Одним из самых

больших недостатков данной библиотеки это нереализванная возможность нормировки данных на максимальное значени, в результате этого нам пришлось реализовать этот функционал собственноручно.

ГЛАВА 3.ОСНОВНЫЕ СВЕДЕНИЯ О СОЗДАНИИ ОДНОСТРАНИЧНЫХ WEB-ПРИЛОЖЕНИЙ С ПОМОЩЬЮ ФРЕЙМВОРКА ANGULARJS

**3.1. AngularJS и совместимые языки программирования**

Для создания веб-приложений в настоящее время существует множество фреймворков: vue.js, react.js, AngularJS и т.д. Все вышеперечисленные фреймворки имеют схожие функции. Они были спроектированы для того, чтобы избавить разработчика от большого колличесва рутинной работы и существенно облегчить разработку[https://habr.com/ru/post/476424/]. В данной работе был выбран AngularJS в силу использования им MVC (Model-View-Controller) шаблона, а также существование двухстороннего связывания, позволяющего динамически изменять и передавать данные из одного места веб-приложения в другое, таким образом AngularJS синхронизирует модель и представление.

AngularJS является JavaScript фреймворком, как видно из названия, главным языком программирования является JavaScript для динамической части приложения и HTML/CSS для разметки. JavaScript - это язык, программы которого выполняемы в разных средах. В данной работе речь будет идти о браузере и серверной платформе Node.js.

В данной работе кроме языка JavaScript также используется “язык гипертекстовой разметки” (HyperText Makup Language = HTML). Этот язык программирования является стандартизованным языком разметки веб-страниц. Этот язык интерптретируется браузерами и, полученный в результате текст, отображается на дисплее. Еще одиним используемым языком программирования в данной работе является СSS (Cascading Style Sheets = каскадные таблицы стилей). Это язык, описывающий внешний вид документа, написанного с помощью стандартизованного языка разметки веб-страниц (HTML).

**2.2. СРЕДА РАЗРАБОТКИ WEBSTORM IDE**

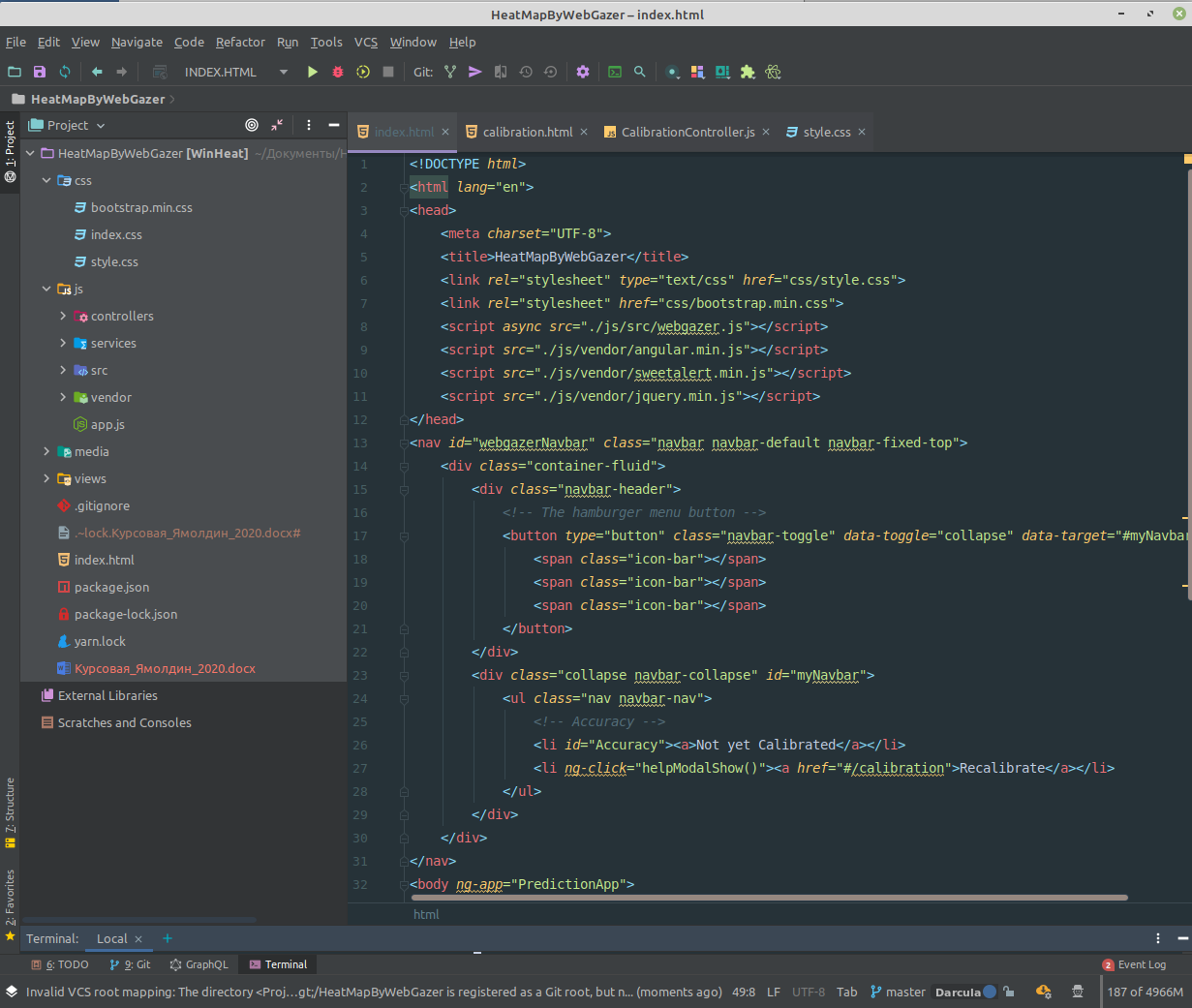


Рис. 1. Стартовое окно WebStorm IDE.

Интегрированная среда разработки WebStorm (IDE) - это кроссплатформенное приложение (для Windows, macOS, Linux), написанное на языке программирования Java.

Исходный код для IDE выпущен под проприетарной лицензией от компании JetBrains. Важно отметить, что для студентов и преподавателей лицензия предоставляется абсолютно бесплатно. WebStorm IDE поддерживает языки JavaScript, CSS и HTML, используя специальные правила структурирования кода. Webstorm IDE имеет LiveEdit - возможность отслеживать “на лету” изменения на странице браузера при редактировании HTML, JavaScript или CSS кода проекта.

**3.3. СТРУКТУРА ПРОГРАММ В СРЕДЕ WEBSTORM IDE ДЛЯ ФРЕЙМВОРКА ANGULARJS.**

Программы в среде программирования WebStorm IDE называются проектами. Проекты, сделанные на AngularJS называются одностраничными придлэениями (SPA = Single Page Application). Приложения AngularJS используют паттерн модель-представление-контроллер (MVC = Model-View-Controller). Кмпонент Model - обычно обозначает сущность, т.е. объект с набором характеристик, обычно это файлы с данными на языке программирования JavaScript. Компонент View - файлы представления. Часто нам нужно предоставить информацию, содержащуюся в компоненте Model на обозрение пользователя, для этого во фреймворке AngularJS используется компонент View, который отображает файлы, написанные на языке программирования HTML на дисплей. Компонент Controller - управляет потоком данных в приложении. Контроллеры взаимодействуют с данными модели и могут в зависимости от поставленных задач изменять их.

AngularJS приложения имеют общую структуру расположения файлов и папок, следование которой является примером хорошего тона в среде программистов- разработчиков.

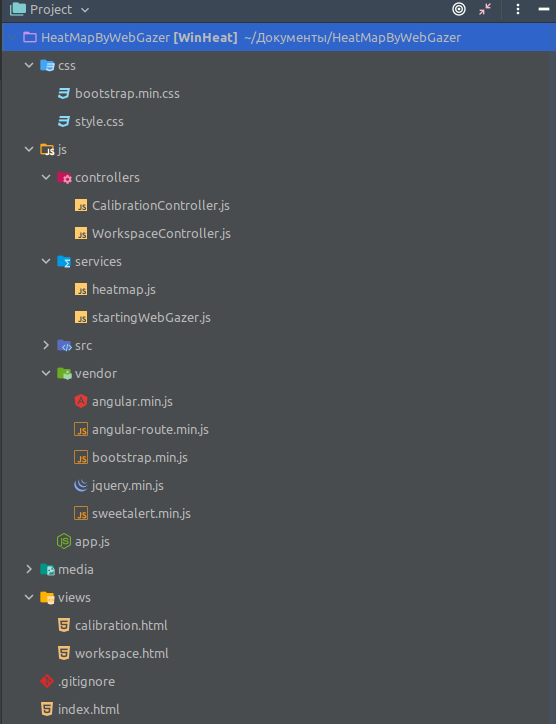


Рис.2. Общая структура приложения.

Файлы AngularJS приложения располагаются в некоторой общей директории, которая называется корневой. На рис.2 она выделена синим цветом и называется HeatMapByWebGazer. Внутри корневой директории располагаются поддиректории: css - директория, содержащая файлы, написанные на языке таблиц стилей и предназначенные для позиционирования и придания стилей для любого элемента из файлов компоненты View, написанных на HTML, js - директория, содержащая все файлы для нашего приложения, написанные на язые JavaScript, является корневой для app.js файла внутри которого мы определяем AngularJS приложение, а также для поддиректорий cotrollers - включающей контроллеры, используемые в нашем приложении, services - включающей особые объекты или функции, выполняющие некоторые общие для всего приложения задачи, src - директория, включающая в себе любые дополнительные JavsScript файлы, которые мы хотели бы использовать в приложении, vendor (поставщик) - директория, содержащая непосредственно тело AngularJS приложения, а также некоторые дополнительные модули и системные файлы, используемые в приложении. Директория media - содержит мультимедиа файлы для нашего приложения, такие как видео, изображения, музыкальные файлы и т.п. Views - директория, содержащая HTML файлы представления приложения. Файл .gitignore - гит файл, опреляющий файлы исключения для гит. index.html - основной файл представления, который первый исполняется при загрузке страницы, внутри него определено AngularJS приложение, подключены внешние срипты, а также внутри которого и происходит основное действие, по-сути, мы всегда находимся внутри этого файла, меняя только его внутренности, посредством взаимодействия с данными и представлениями с помощью AngularJS фреймворка. Именно поэтому приложения, написанные с помощью AngularJS являются SPA.

**ГЛАВА 4. ПРАКТИЧЕСКАЯ ЧАСТЬ**

**4.1 Подготовка к работе приложения в index.html**

Как было сказано выше, файл index.html выполняется перым при запуске AngularJS приложения, поэтому важно внутри него подключить все внешние сторонние срипты, тело AngularJS приложения и файлы стилей (рис. 2.) , которые мы собираемся использовать в нашем приложении, а также используемые внутри него сервисы и контроллеры (рис. 3.).

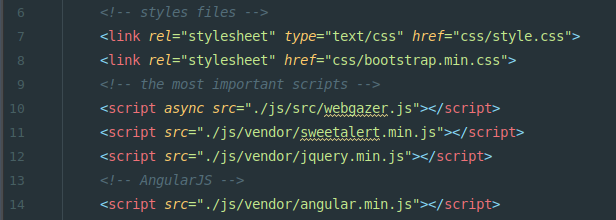


Рис. 2. Сторонние скрипты, файлы стилей и тело AngularJS.

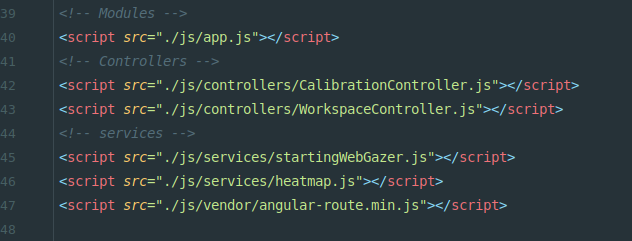


Рис. 3. Определение приложения, его сервисов и контроллеров.

**4.2 Создание AngularJS приложения**

Файл app.js является обязательным файлом AngularJS приложения.



Рис. . Вид файла app.js

Этот файл “говорит” браузеру, что это приложение AngularJS, имеющее названиe PredictionApp. В квадратных скобках указываются зависимости для данного приложения, иными словами, указываются некоторые дополнительные модули (функции), которых изначально нет в теле AngularJS приложения. Ниже пишется конфигурация приложения. В зависимости мы указали модуль ‘ngRoute’ который находится в файле angular-min.js и позволяет использовать маршрутизацию внутри нашего приложения. Для конфигурации маршрутов используется объект $routeProvider, определенный внутри модуля ngRoute. Метод $routeProvider.when принимает два параметра: название маршрута и объект маршрута. Название маршрута это URL, по которому осуществляется доступ к представлению, находящемуся в templateUrl: и обрабатывающего его котроллеру в controller:. Метод $routeProvider.otherwise вызывается в том случае, если приложению не удалось найти сопоставить маршрут из $routeProvider.when и браузерным запросом и перенаправляет пользователя по маршруту, прописанному внутри redirectTo:.