Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра

Проектирования информационно-компьютерных систем

“Приложение по расчёту надёжности радиоэлементов”

|  |  |
| --- | --- |
| Проверил:  Шнейдеров Е.Н. | Выполнили  ст. гр. 313801:  Апанович В.С.  Коваль А.В.  Мисюля А.А. |

Минск, 2015

|  |  |
| --- | --- |
| **Содержание** | |
|  |  |
| 1 Выбор программного обеспечения …….…………..……….….………..... | 2 |
| 2 Среда разработки и фреймворки……….…………………………….......... | 2 |
| 3 Описание клиент-серверного взаимодействия…………………................ | 2 |
| 3.1 Запрос на авторизацию ………………………………….……….……. | 3 |
| 3.2 Запрос на загрузку проекта(ов)…..…………………………………….  3.3 Запрос на сохранение проекта…………………………………………  3.4 Запрос на получение всех свойств……………………………………..  3.5 Запрос на получение готовых элементов……………………………...  3.6 Запрос на выход из аккаунта пользователя…………………………... | 4  4  5  6  6 |
| 4 Настройка HTTPS….………………………………………………….….....  4.1 Создание хранилища (keystore)………………………………………...  4.2 Конфигурация SSL-коннектора……………………………………….. | 6  6  7 |

5 Описание клиентской части приложения…………………………………. 7

5.1 Задачи, решаемые приложением………………………………………. 7

5.2 Функции, доступные пользователю………………………………….... 7

5.3 Структура проекта……………………………………………………….

5.4 Описание интерфейса программы……………………………………....

5.5 Описание технической реализации……………………………………..

5.6 Результат разработки ……………………………………………………

**1 Выбор программного обеспечения**

В качестве серверного ПО был выбран apache-Tomcat 8.0.20. Он был выбран в первую очередь из-за бесплатности и удобности в использовании.

**2 Среда разработки и фреймворки**

Для разработки серверной части в качестве среды разработки была выбрана Intellij Idea 14.1.2 Ultimate, это было сделано потому что эта среда, по нашему мнению, является лучшей для разработки на языке Java. Фреймворки не были использованы.

Дополнительные библиотеки: javax.servlet (для непосредственного создания сервлетов), org.json (для создания и обработки JSON-объектов) и jersey 1.19 (для http-взаимодействия с базой данных по REST API), neo4j-rest-graphdb-2.0.1(подключение к базе данных).

Для разработки клиентской части приложения был использован языки javaScript, HTML, CSS. В качестве среды разработки был выбран текстовый редактор SublimeText3, т.к. он распространяется бесплатно и обладает большим набором доступных расширений, позволяющих упростить работу при разработке. В процессе разработки был использован фреймворк AngularJS v1.3.15. Выбор данного фреймворка обусловлен широким спектром возможностей, которые он способен предоставить разработчику.

**3 Описание клиент-серверного взаимодействия**

Поддерживаются два способа передачи параметров:

1. Методы **GET**: вводятся в URL запроса.  
   Пример: *?<param\_name>=<param\_value>*
2. Методы **POST**: вводятся в теле HTTP запроса.  
   Пример: *?<param\_name>=<param\_value>*

Все запросы в URI-формате имеют следующий вид:

https://<server>/<projectFolder>/<method name>/?<get params>

Где:

* *<server>* — адрес сервера вида <IP-address:port number>. IP-address изменяется динамически, port number – 8080 или 8443(предпочтительно). Пример:

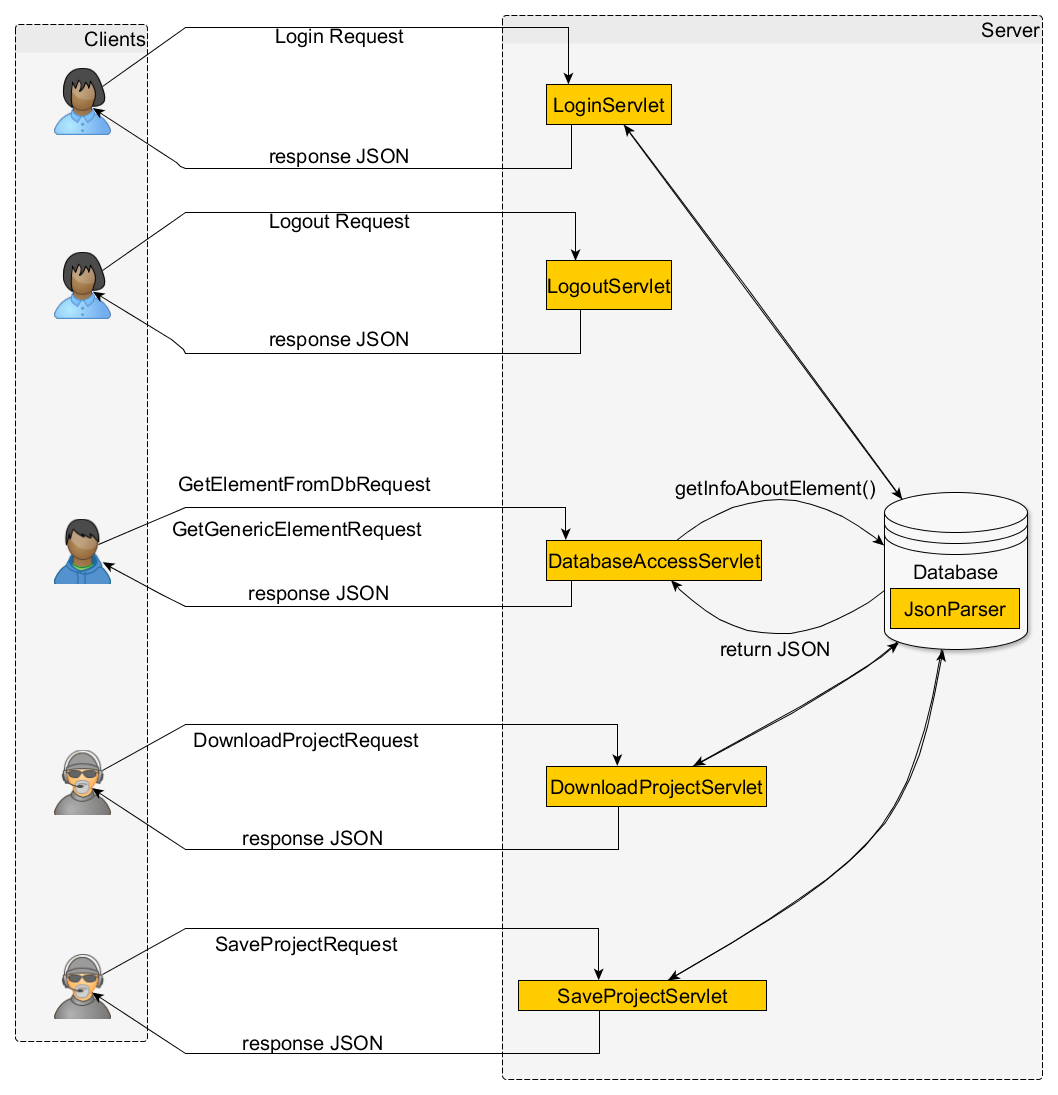
<server> = 192.168.1.2:8443

* *<projectFolder>* — папка, в которой находится сервер. Использовать <projectFolder> = arion
* *<method name>* — поддерживаются следующие варианты: login, logout, arion, download, save. Ниже будет подробно описан каждый из них.
* *<get params>* — параметры метода GET для запроса

Любые текстовые параметры необходимо передавать в кодировке UTF-8.

Данные всегда возвращаются в формате JSON.

Информация, получаемая из базы данных, избыточна. Класс JsonParser.class служит для обработки и приведения полученной информации к удобному для дальнейшей работы формату. Далее везде опущено.



**3.1 Запрос на авторизацию - LoginRequest (метод GET)**

Запрос:

/login?ul=<userLogin>&up=<userPassword>

Запрос приходит в LoginServlet.class, из него берётся userLogin и userPassword. Проверяется, есть ли такой пользователь в базе данных и правильно ли введены логин и пароль. Если такой пользователь есть, то создаётся сессия с именем пользователя. Далее во всех классах идёт проверка на авторизацию(т.е. если сессии с именем пользователя не существует, то он не сможет получать данные от сервера).

Ответ:

а) Введены верные данные

{"data":{"auth":"true","description":"Авторизация пройдена"}}

б) Введены неверные данные

{"data":{"auth":"false","description":"Ошибка авторизации. Введены неверные данные"}}

**3.2 Запрос на загрузку проекта(ов) - DownloadProjectRequest (метод GET)**

Запрос:

а) /download?all=true

Запрос приходит в DownloadProjectServlet.class, далее

идёт запрос в базу данных и из неё выбираются только имена проектов пользователя

Ответ:

а) Сервер возвращает массив имен пользовательских проектов {"data":[{"name":"name1"},{"name":"name2"},{"name":"name3"},…,{"name":"nameN"}]}

Запрос:

б) /download?name=<projectName>

Запрос приходит в DownloadProjectServlet.class, далее идёт запрос в базу данных по параметру projectName. После чего возвращается элемент с именем projectName следующего вида:

Ответ:

б) Сервер возвращает конкретный пользовательский проект

{"data":{"someKey":"someValue"},"name":"projectName"}

**3.3 Запрос на сохранение проекта - SaveProjectRequest (метод POST)**

Запрос:

на URL /save присылается POST-запрос, тело которого представляет собой JSON вида {"data":{"someKey":"someValue"},"name":"projectName"}. Поле “data” – JSON-описание сохраняемого проекта.

Этот запрос попадает в SaveProjectServlet.class, после чего идёт запрос к базе данных, в ней создаётся новый узел с данными : {"data":{"someKey":"someValue"},"name":"projectName"}. Клиенту возвращается ответ вида:

Ответ:

{"data": "Проект успешно сохранен"}

**3.4 Запрос на получение всех свойств, описывающих выбранную группу элементов –GetGenericElementRequest (метод GET)**

Запрос:

/arion?cn=<className>&gn=<groupName>&mt=<methodType>

<className>=Коммутационные изделия/Конденсаторы

<groupName>=Микропереключатели/Слюдяные

<methodType>=Отечественная методика

Этот запрос попадает в DatabaseAccessServlet.class, из него берутся параметры className, groupName, methodType. После этого идёт запрос к базе данных, в ней происходит поиск по параметрам, после чего выдаётся ответ:

Ответ:

{    
  "data":{    
    "method":"Lb\*Kp\*Kc\*Ke\*Kq",  
    "coefficients":[    
      {    
        "name":"Коэффициент приёмки",  
        "formula":"Q",  
        "key":"Kq"  
      }, …

 ],  
    "properties":[    
      {    
        "default":"5",  
        "min":"0",  
        "visible":"true",  
        "max":"1000000000000",  
        "name":"Ёмкость конденсатора, пФ",  
        "type":"2",  
        "key":"С"  
      },

…  ]  
  }  
}

**3.5 Запрос на получение готовых элементов из БД - GetElementFromDbRequest (метод GET)**

Запрос:

/arion?ge=<groupName>

<groupName>=Слюдяные

Этот запрос попадает в DatabaseAccessServlet.class, из него берётся параметр groupName. После этого идёт запрос к базе данных, в ней происходит поиск готовых элементов по параметру, после чего выдаётся ответ:

Ответ:

{"data":[{"name":"КСОТ","properties":[{"default":"3E-08","name":"Базовая интенсивность отказов"}]},{"name":"К31-7","properties":[{"default":"3E-08","name":"Базовая интенсивность отказов"}]},{"name":"К31-14","properties":[{"default":"1E-08","name":"Базовая интенсивность отказов"}]}]}

**3.6 Запрос на выход из аккаунта пользователи - LogoutRequest (метод GET)**

Запрос:

/logout

Этот запрос попадает в LogoutServlet.class. В этом классе берется текущая сессия и удаляется( т.е. у пользователя нет прав на получение данных от сервера). Пользователю приходит ответ:

Ответ:

{"data":{"auth":"false","description":"Вы вышли из профиля"}}

**4 Настройка HTTPS**

Для защиты передаваемых данных был использован протокол HTTPS (HyperText Transfer Protocol Secure).

**4.1 Создание хранилища (keystore) с ключом**  
Наберем в командной строке следующий код:

>keytool -genkey -alias tomcat -keyalg RSA -keystore mystore -validity 999 -keysize 512

Здесь:  
— tomcat — имя псевдонима  
— keyalg — алгоритм формирования ключа  
— keystore — имя хранилища  
— validity – Срок действия сертификата  
— keysize – Размер ключа

После завершения «диалога» с консолью создается хранилище с введенными параметрами.

**4.2 Конфигурация SSL-коннектора** (фрагмент **server.xml**):  
Находим запись

<!-- Define a SSL HTTP/1.1 Connector on port 443

This connector uses the JSSE configuration, when using APR, the

connector should be using the OpenSSL style configuration

described in the APR documentation -->

и ниже нее добавляем

<Connector port="8443" SSLEnabled="true" maxHttpHeaderSize="8192"

maxThreads="150" minSpareThreads="25" maxSpareThreads="200"

enableLookups="false" disableUploadTimeout="true"

acceptCount="100" scheme="https" secure="true"

clientAuth="false" sslProtocol="TLS"

keystoreFile="/conf/mystore" keystorePass="11081996" keystoreType="JKS"

keyAlias="tomcat" useBodyEncodingForURI="true"/>

Находим строку:

<Listener className="org.apache.catalina.core.AprLifecycleListener" SSLEngine="on" />

— по умолчанию 27 строка, закомментируем её.  
Также комментируем следующую строку:

<Listener className="org.apache.catalina.mbeans.ServerLifecycleListener" />

Теперь переносим файл **mystore** из папки с java в папку /tomcat/conf. Запускаем Tomcat, SSL- шифрование уже работает.

**5. Описание клиентской части приложения**

**5.1 Задачи, решаемые приложением**

Основная задача, которую решает приложение – это создание модели электрического устройства из доступных в базе данных компонентов и последующий расчет надежности данного устройства, сделанный на основе данных о надежности каждого из компонентов устройства.

**5.1 Функции, доступные пользователю**

С точки зрения пользователя, ему доступны следующие функции:

- авторизация на сервере

- создание модели устройства

- добавление в устройство модулей, которые могут содержать в себе как непосредственно электронные компоненты, так и другие модули

- добавление в устройство электронных компонентов

- определение технических характеристик электронных компонентов

- расчет надежности электронного компонента с заданными параметрами

- редактирование модели (формулы) по которой производится расчет надежности

- поиск элемента по имени и позиционному обозначению

- отображение структуры устройства в виде древовидной структуры

- отображение списка всех используемых элементов

- расчет интенсивности отказа для выбранного модуля, в том числе, расчет для всего устройства

- редактирование названия элемента, его позиционного обозначения

- удаление созданного элемента или модуля из устройства

**5.3 Структура проекта**

│ .gitignore

│ index.html

│ README.md

│ tree.txt

│

├───css

│ │ main.css

│ │

│ ├───bootstrap

│ │ bootstrap.min.css

│ │

│ ├───fonts

│ │ │ ionicons.eot

│ │ │ ionicons.svg

│ │ │ ionicons.ttf

│ │ │ ionicons.woff

│ │ │

│ │ └───cuprum

│ │ Cuprum-Bold.ttf

│ │

│ └───ionic

│ ionic.css

│ ionicons.min.css

│

├───debug

│ bugList.txt

│

├───docs

│ JsonAnswerElementFromDB.json

│ jsonAnswerExample\_editedByKav.json

│ JsonAnswerGenericElement.json

│ JsonAnswerUserAuthorisiation.json

│ taskDescription.txt

│

├───images

│ favicon.ico

│ texture.png

│

├───js

│ app-core.js

│ app-editor.js

│ app-header.js

│ app-i18n.js

│ app-output.js

│ app.js

│

├───libs

│ ├───angular

│ │ angular-route.js

│ │ angular-route.min.js

│ │ angular.js

│ │ angular.min.js

│ │

│ ├───angular-translate

│ │ angular-translate.js

│ │ angular-translate.min.js

│ │

│ ├───bootstrap

│ │ bootstrap.min.css

│ │

│ ├───ionic

│ │ ionic.bundle.js

│ │ ionic.bundle.min.js

│ │

│ └───mathjs

│ math.min.js

│

├───resources

│ answer.json

│ capacitors.json

│ capacitorsWorkable.json

│ Classifier.xml

│ elements.json

│ micro.json

│ microWorkable.json

│ trialAnswer.json

│ undoneAnswerRC.json

│

├───templates

│ addElementModal.html

│ addModuleModal.html

│ authorizationForm.html

│ editPanel.html

│ header.html

│ mainView.html

│ modal.html

│ nodeEditor.html

│ outputPanel.html

│ searchElementModal.html

│ treeItem.html

│ treePanel.html

│

└───temporary

│ a01.sublime-project

│ a01.sublime-workspace

│ bootstrap-3.1.1-dist.zip

│ bootstrap-3.3.4-dist.zip

│ jsonFull.json

│ seamless\_paper\_texture.zip

│ ui-bootstrap-tpls-0.13.0.js

│ ui-bootstrap-tpls-0.13.0.min.js

│

├───bootstrap-3.1.1-dist

│ └───bootstrap-3.1.1-dist

│ ├───css

│ │ bootstrap-theme.css

│ │ bootstrap-theme.css.map

│ │ bootstrap-theme.min.css

│ │ bootstrap.css

│ │ bootstrap.css.map

│ │ bootstrap.min.css

│ │

│ ├───fonts

│ │ glyphicons-halflings-regular.eot

│ │ glyphicons-halflings-regular.svg

│ │ glyphicons-halflings-regular.ttf

│ │ glyphicons-halflings-regular.woff

│ │

│ └───js

│ bootstrap.js

│ bootstrap.min.js

│

├───bootstrap-3.3.4-dist

│ └───bootstrap-3.3.4-dist

│ ├───css

│ │ bootstrap-theme.css

│ │ bootstrap-theme.css.map

│ │ bootstrap-theme.min.css

│ │ bootstrap.css

│ │ bootstrap.css.map

│ │ bootstrap.min.css

│ │

│ ├───fonts

│ │ glyphicons-halflings-regular.eot

│ │ glyphicons-halflings-regular.svg

│ │ glyphicons-halflings-regular.ttf

│ │ glyphicons-halflings-regular.woff

│ │ glyphicons-halflings-regular.woff2

│ │

│ └───js

│ bootstrap.js

│ bootstrap.min.js

│ npm.js

│

└───seamless\_paper\_texture

├───seamless\_paper\_texture

│ readme.txt

│ seamless\_paper\_texture.png

│

└───\_\_MACOSX

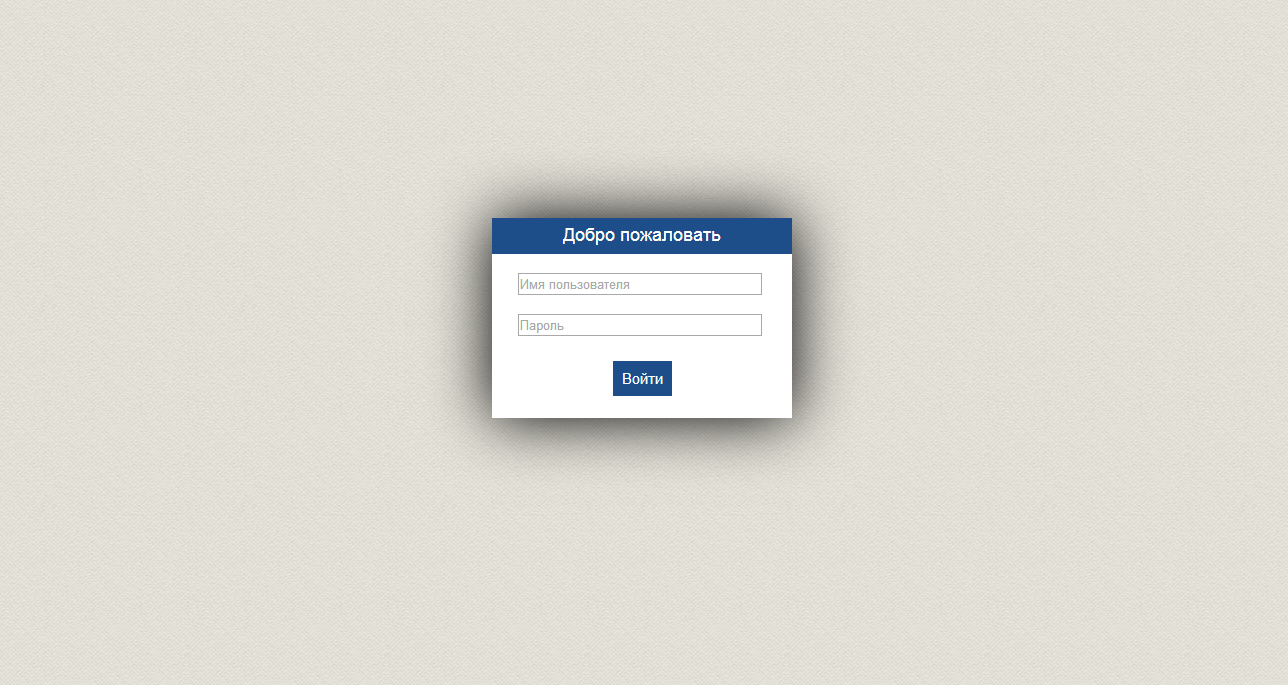
└───seamless\_paper\_texture

.\_readme.txt

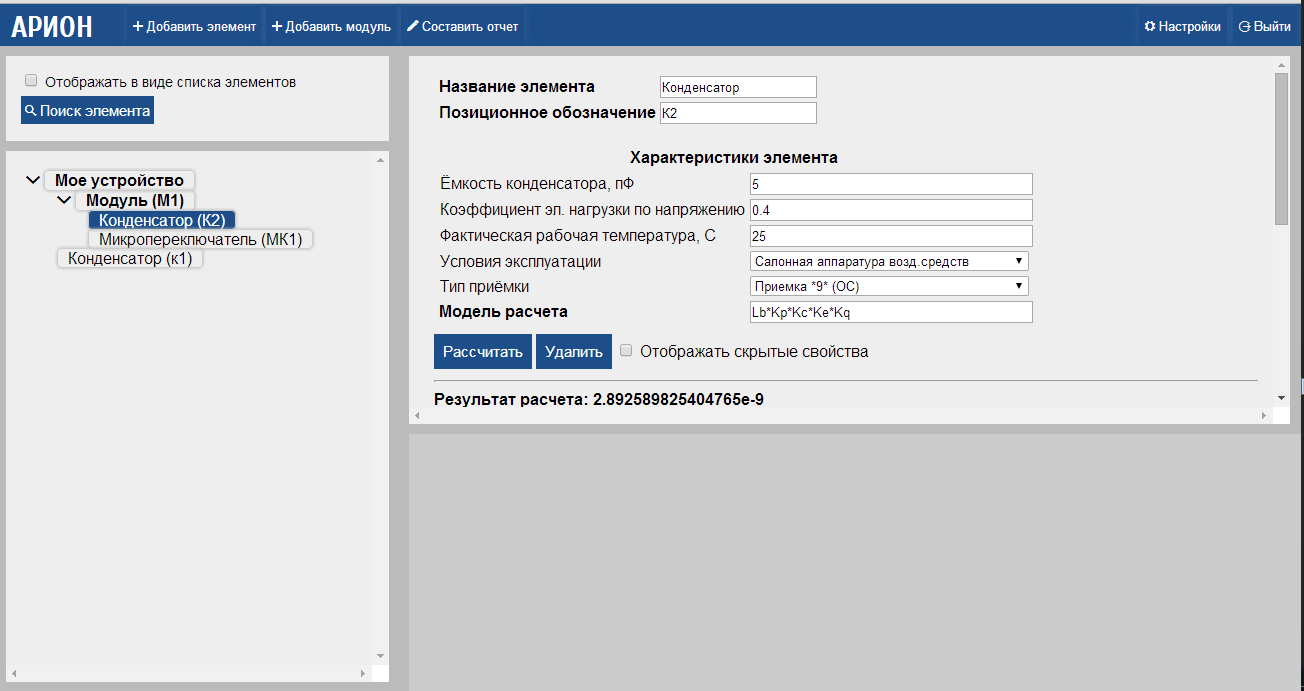
.\_seamless\_paper\_texture.png

**5.4 Описание интерфейса программы**

При первичном запуске приложения, пользователю отображается форма ввода регистрационных данных:



После ввода регистрационных данных осуществляется переход на главную страницу приложения.



На странице можно выделить следующие зоны:

-верхняя строка-меню.

Обеспечивает визуализацию базовых функций приложения, такие как добавление компонентов и расчет устройства.

- область отображения дерева

Здесь отображается структура устройства в виде дерева или в виде списка элементов. Имеется кнопка поиска по элементам и переключатель режима отображения

- область редактирования параметров

Здесь отображаются характеристики, доступные для данного узла. Также здесь имеются кнопки для выполения расчета и удаления узла. В нижней части области выводятся резльтаты расчета

- область построения графиков

Зарезервирована для будущего использования

**5.4 Описание технической реализации.**

Исходный код приложения располагается в папке /js.

Логика программы построена на основе шаблона MVC. В частности, касательно данной программы:

- Model – данные о структуре устройства, данные о свойствах компонентов устройства, данные о состоянии приложения и сервисы, обрабатывающие эти данные

-View – набор независимых HTML-шаблонов, подключаемых асинхронно в процессе работы программы.

- Controller – бизнес-логика приложения. Обрабатывает данные модели (Model) и связывает их с отображением (View)

В качестве модели используются данные json-файлов (получаемых с сервера и хранящихся локально), а также json-объекты, генерируемые программой в процессе работы – данные о состоянии приложения.

Данные о доступных пользователю для выбора компонентах хранятся в файле /resources/elements.json . В качестве обработчика уровня модели для этих данных выступает сервис elementSelectionService (app-core:91) – этот сервис позволяет считать данные из файла и выделить оттуда классы элементов, методики расчета и группы элементов. Для доступа к файлу используется http.get – request

Данные о свойствах компонентов содержатся в json-конструкциях, присылаемых с сервера в ответ на клиентский запрос, содержащий информацию о выбранном компоненте. Обработчиком уровня модели выступает databaseService (app-core:129), который отправляет серверу запрос и расширяет объекты, представляющие электрические компоненты, необходимыми свойствами.

Данные о пользовательской модели устройства хранятся в программном json-объекте, по сути, представляющим собой древовидную структуру модулей-узлов и элементов-листьев. Корнем дерева является узел root, видимый пользователю как «Моё устройство». Обработчик этих данных – сервис treeDataService (app-core:6). Этот сервис предоставляет интерфейсы для таких команд как добавление узла, добавление листа, удаление листа, поиск по дереву элементов.

В качестве View-компонента выступают следующие файлы:

addElementModal.html – шаблон для всплывающего окна (модала), возникающего при добавлении нового элемента в устройство

addModuleModal.html – шаблон для модала, возникающего при добавлении нового модуля в устройство

authorizationForm.html – шаблон для формы авторизации

editPanel.html – шаблон для области редактирования параметров компонета

header.html – шаблон для верхней строки-меню

mainView.html – корневой шаблон, определяющий контент области отображения. Содержит в себе директивы со ссыллками на другие шаблоны и условиями, при которых их необходимо показывать

modal.html – базовый шаблон для модала

outputPanel.html – шаблон для панели вывода. Зарезервирован для отображения графиков и диаграмм

treeItem.html – шаблон, определяющий отображение узлов дерева

treePanel.html – шаблон, определяющий внешний вид области отображения дерева.

В качестве Controller-компонента выступают контроллеры приложения.

Корневым контроллером приложения является RootCtrl (app-core:192). В нем содержится ссылка на дерево элементов, переменные, хранящие данные основных переключателей. RootCtrl определен в качестве контроллера, который управляет директивой mainView, таким образом, все вложенные директивы имеют доступ к данным, хранящимся в RootCtrl.

AddElementController связывает данные модели файла elements.json и шаблон addElementModal.html. В контроллере определены функции, вызываемые по нажатии на кнопки, расположенные во всплывающем меню добавления элемента.

HeaderCtrl реализует функции, вызов которых доступен по нажатии на кнопки верхней строки-меню. Связан с шаблоном header.html

TreeCtrl обеспечивает функционал, определяющий работу с деревом. Использует сервис treeDataService и связан с шаблоном treePanel.html

Модуль app-i18n обеспечивает функционал, необходимы для локализации приложения. Включает в себя словарь для русской версии приложения, предпочтительным языком по умолчанию является русский. Работает на основе расширения для фреймворка AngularJS angular-translate. Является потенциальной точкой расширения в случае использования приложения на рынке, с языком отличным от русского.

ChartCtrl является потенциальной точкой расширения для подключения графиков.

EditorCtrl обеспечивает функционал по редактированию параметров выбранного компонента. Использует treeDataService для отображения данных о характеристиках элемента и mathService для вызова функции расчета модели.

Так же в приложении есть служебные сервисы, обеспечивающие работу контроллеров. К таким сервисам относится mathService. Его основным назначением является проведение расчета на основе данных для расчета (в формате JSON-объекта) и формулы для расчета в строковом формате.

Пример объекта, содержащего данные для расчета:

{    
    **"C"**:5,  
    **"I"**:0.4,  
    **"H"**:3,  
    **"S"**:0.4,  
    **"G"**:1,  
    **"N"**:358,  
    **"B"**:16,  
    **"A"**:9.885e-8,  
    **"Tn"**:25,  
    **"L"**:4e-8,  
    **"X"**:4,  
    **"Q"**:0.3,  
    **"Kq"**:0.3,  
    **"Ke"**:4,  
    **"Kc"**:0.5010900647302042,  
    **"Kp"**:0.12026238848242224,  
    **"Lb"**:4e-8  
}

mathService в процессе работы совершает следующие операции:

- осуществляет проверку формулы на валидность, на основе расстановки скобок

- осуществляет проверку формулы на безопасность, путем проверки наличия в строке ключевых слов языка программирования javaScript

- осуществляет замену разделителей-запятых на точки

- осуществляет подстановку необходимых значений переменных

- осуществляет замену математических операций, описанных в формально-математическом формате, на формат, пригодный для выполнения программой

- осуществляет непосредственно расчет значений.

**5.6 Результат разработки**

По результатам разработки было создано работающее веб-приложение, позволяющее производить расчет надежности электронного устройства, создаваемого пользователем. Был реализован базовый набор функций, обеспечивающий стабильную работу приложения. Потенциальными направлениями развития проекта являются:

- добавление возможности построения графического материала по результатам расчета.

- добавление возможности сохранения/загрузки/открытия пользовательских проектов на сервер/локальное устройство

- поддержание возможности формирования отчетов

- поддержание работы с несколькими устройствами

- локализация приложения для не русскоязычного рынка

- поддержка минификации кода

Потенциальными уязвимостями программы являются:

- небезопасное хранение данных авторизации

- возможность внедрения пользовательского кода

- отсутствие механизмов шифрования/хеширования