Министерство транспорта Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Российский университет транспорта» (РУТ (МИИТ)

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

Отчёт

по практике

«Учебная практика»

1. ­ Практика по получению первичных профессиональных умений и навыков, в том числе умений и навыков научно-исследовательской деятельности

Выполнил: ст. гр. ТУУ-211

Дидушок Д.А.

Проверил: доц. Сафронов А.И.

Москва­ – 2019 г.

**Задание 1.**

Цель работы:

• Создание браузерного калькулятора на базе фреймворка "vue.js"

• Разработка пошагового пользовательского интерфеса в рамках одностраничного веб-приложения на базе фреймворка "vue.js"

Задачи:

1.

•Определение понятия "Фреймворк"

•Построение графического пользовательского интерфейса на базе фреймворка

•Определение понятия " iQuery "

•Реализация функций посредством подключения "iQuery" к разработке

•Исключение " iQuery " из проекта и написание всех расчетных функций с применением фреймворка "vue.js"

•Механизм отлова и контроля возникающих ошибок ввода данных

•Внедрение и постановка задачи на разработку аналогичных приложений-калькуляторов в рабочую программу Учебной практики, реализуемой распределенно в течении семестра на кафедре и информационно-вычислительном центре ИТТСУ

2.

•Разработка одностраничного веб-приложения, в котором информация, хранимая в текстовом файле записана в формате json

•Построить приложение по принципу пошагового веб-интерфейса

•Декомозиция кода

•Переработка и адаптация фрагмента кода под взаимодействие с фреймворком «Vue.js»

•Разработка фреймов веб-приложения

• Разработанное одностраничное веб-приложение применять для изучения основ программирования на JavaScript на базе фреймворка Vue.js. Внедрение результатов поставленной задачи в курс Учебной практики, практики по приобретению первичных профессиональных умений и навыков, в том числе умений и навыков научно-исследовательской деятельности.

Содержательная часть:

• Текст доклада:

**ВОЗМОЖНОСТИ ОДНОСТРАНИЧНЫХ ВЕБ-ПРИЛОЖЕНИЙ: БРАУЗЕРНЫЙ КАЛЬКУЛЯТОР НА БАЗЕ ФРЕЙМВОРКА «VUE.JS»**

В ходе работ, проводимых на кафедре «Управление и защита информации» и связанных с интеграцией тематики веб-программирования в состав рабочих программ учебных дисциплин по курсам алгоритмизации и технологий программирования, а также информационного обеспечения систем управления был создан наглядный и функциональный пример, позволяющий обучающимся освоить на практике основы построения одностраничных веб-приложений.

По результатам анализа рынка разработки современного программного обеспечения выбран фреймворк «Vue.js». В качестве критерия отбора выступило максимальное количество переходов к использованию «Vue.js» разработчиков из различных организаций в единицу времени по всему миру.

Фреймворк – это заготовки, шаблоны для программной платформы, определяющие структуру программной системы. Иначе: программное обеспечение, облегчающее разработку и объединение разных модулей программного проекта.

В разделе образовательных проектов, способствующих изучению фреймворка «Vue.js» и рекомендованных его разработчиком (Эваном Ю), был найден пример стильного калькулятора, в котором на базе упомянутого фреймворка построен только графический пользовательский интерфейс, а все функции реализованы посредством подключения «jQuery» к разработке.

«jQuery» – является библиотекой, написанной на сценарном языке JavaScript. Исходно она предназначена для взаимодействии JavaScript и HTML. Библиотека «jQuery» помогает получать доступ к любому элементу DOM (Document Object Model – «объектной модели документа» – независящего от платформы и языка программирования интерфейса, позволяющего программам и сценариям получить доступ к содержимому

HTML-, XHTML- и XML-документов, а также изменять содержимое, структуру и оформление таких документов). Библиотека также позволяет обращаться к атрибутам и содержимому элементов DOM, манипулировать ими.

Авторы поставили перед собой задачу исключения «jQuery» из проекта и написания всех расчётных функций полностью с применением фреймворка «Vue.js». Такое решение основано на общей концепции повышения надёжности программной системы, в которой доказано утверждение, что система обладает большей надёжностью и запасом по устойчивости тогда и только тогда, когда в ней минимизировано количество подключённых внешних модулей. Поскольку фреймворк «Vue.js» тоже является, в некотором частном смысле, JavaScript библиотекой, то при исключённой «jQuery» показатели по надёжности программной системы возрастают.

В процессе модернизации стильного калькулятора выяснилось, что некоторые графические элементы теряют свою устойчивость по положению при превышении некоторого наперёд известного количества вводимых символов (они определены опытным путём). Так был предусмотрен механизм отлова и контроля возникающих ошибок ввода данных. Любая выявленная ошибка индицируется в поле вывода ответа словом «Ошибка». Выводом из состояния ошибки является подача любой новой команды калькулятору или нажатие на кнопку «Сброс».

Разработанный калькулятор является реактивным одностраничным веб-приложением. С его помощью можно быстро и наглядно рассказать обучающимся о событиях, обрабатываемых в рамках объектно-ориентированного подхода к программированию, а также о связанных с этими событиями программных методах.

Авторами планируется внедрение и постановка задачи на разработку аналогичных приложений-калькуляторов в рабочую программу Учебной практики, реализуемой распределённо в течение семестра на кафедре и информационно-вычислительном центре ИТТСУ.

На сегодня актуальность приобретения навыков веб-программирования обучающимися довольно высока, поскольку большинство дочерних предприятий ОАО «РЖД» занято разработкой именно одностраничных веб-приложений для своего основного заказчика.

**Пошаговый графический пользовательский интерфейс в рамках одностраничного веб-приложения на базе фреймворка «*Vue.js*»**

Работа посвящена подходам к демонстрации функциональных возможностей фреймворка «*Vue.js*» в рамках учебных дисциплин, связанныхс программированием.

Поставлена задачана разработку одностраничного веб-приложения, в котором информация, хранимая в текстовом файле и записанная в формате *json,*была быпредставленапользователю в различных вариациях, таких как:

* исходный текст *json*-файла с разбиением на маркированные строки,
* форматированный текст *json*-файла, представленный в виде нумерованного списка,
* форматированный текст *json*-файла, инкапсулированный в комбинированный список.

Приложение построено по принципу пошаговоговеб-интерфейса, аналогичного по функциональности, например, инсталлятору программного продукта на персональный компьютер, где пользователь шаг за шагом определяет место размещения файлов приложения, а также выполняет детальную настройку возможностей и ограничений программного обеспечения, делая его частью операционной системы.

Разработанное одностраничное веб-приложение прекрасно подходит для изучения основ программирования на *JavaScript*на базефреймворка*Vue.js*. Здесь каждая следующая страница, а лучше сказать, «фрейм» использует более сложные интерфейсные элементы управления и размещения данных по сравнению с предыдущей. Смена фреймов происходит без перезагрузки html-документа, то есть без повторного обращения к сетевому адресу, что существенно экономит сетевой трафик и время взаимодействия оператора с системой.

Также для удобства освоения выбранной технологии программирования применён подход по декомпозиции кода. Логически разделённые фрагменты – фреймы размещены в отдельных файлах с расширением *\*.js* и вызываются по необходимости посредством изменения значения флага, которое реализовано по нажатии на кнопку. Так все кнопки программы связаны с событиями, каждое из которых меняет состояние флага, а вместе с ним реактивно меняется и демонстрируемый оператору фрейм.

Исходно в сети Интернет был найден пример для реализации загрузки сведений из текстового файла на веб-страницу. Примерпредставлен простыми сценарными средствами *JavaScript*. В кратчайшие срокивыполнена переработка и адаптация найденного фрагмента кода под взаимодействие сфреймворком «*Vue.js*».

Далее рассмотреныфреймы созданного веб-приложения:на первом пользователю предложен выбор файла, подлежащего загрузке. После успешной загрузки файла и применения выбора раскрывается второйфрейм, на котором представлено содержимое *json*-файла известной структуры без форматирования. Первый фрейм тем временем скрывается и делается недоступным для пользователя. Со второго фрейма можно вернуться обратно к выбору файла, а также продвинуться «Далее». Продвижение «Далее» переводит приложение к фрейму, на котором вывод организован с форматированием по определённому правилу, представленному ниже:

|  |  |
| --- | --- |
| {{*el.id}}.*<*b*>Название товара:</*b*><*i*>{{*el.productname*}}</*i*><*b*> Пол:</*b*><*i*>{{*el.sex*}}</*i*><*b*>Описание:</*b*><*i*>{{*el.description.effect*}}</*i*><*b*> Цена: </*b*><*i*>{{*el.price*}}</*i*> | (1) |

Указанная конструкция обёрнута в цикл по каждому известному элементу списка. В ней компоненты, указанные внутри усатых скобок «{{}}»–это параметры рассматриваемого в данный момент элемента строки. В угловых скобках указаны *html*-теги, необходимые для форматирования представленного текста. Внутри конструкции <*b*></*b*>содержится жирный текст, внутри конструкции <*i*></*i*>–курсив.

С фрейма возможен переход обратно к выводу сведений без форматирования, а также переход к следующему, завершающемуфрейму с размещённым на ней комбинированным списком.

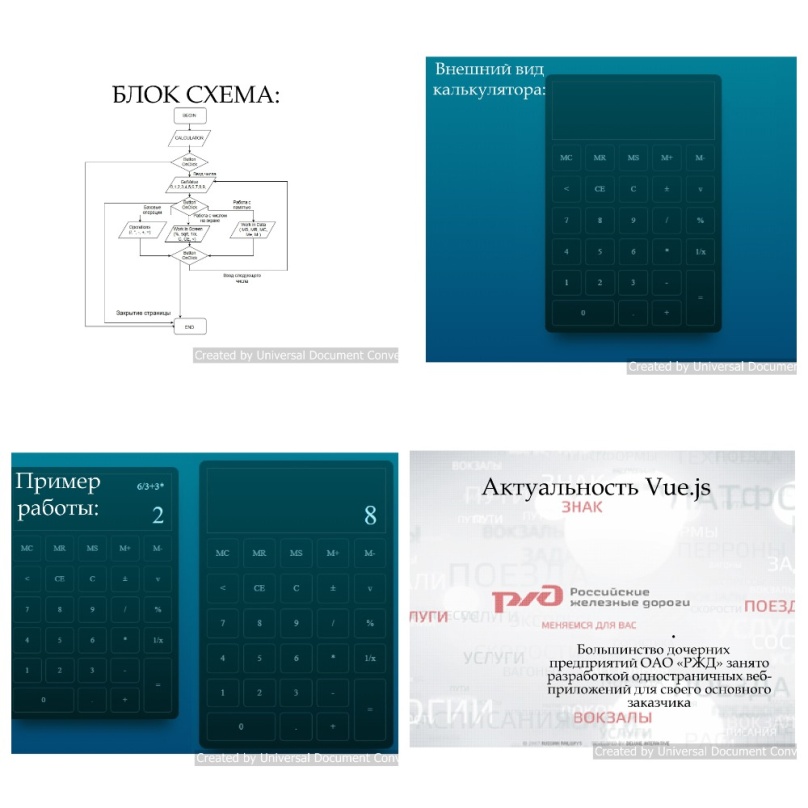
При выборе элемента в комбинированном списке справа от него отображается подробное описание компонента. Указанное становится доступно после размещения на странице нижеследующего фрагмента кода:

|  |  |
| --- | --- |
| <*select v-model="selected"*>  <*option v-for="el in this.$root.fileData"*  *v-bind:value="el.description.effect"*> {{*el.productname*}} </*option*>  </*select*>  <*span v-if="selected!==''"*><b>Описание:</b><i> {{*selected*}} </i></span> | (2) |

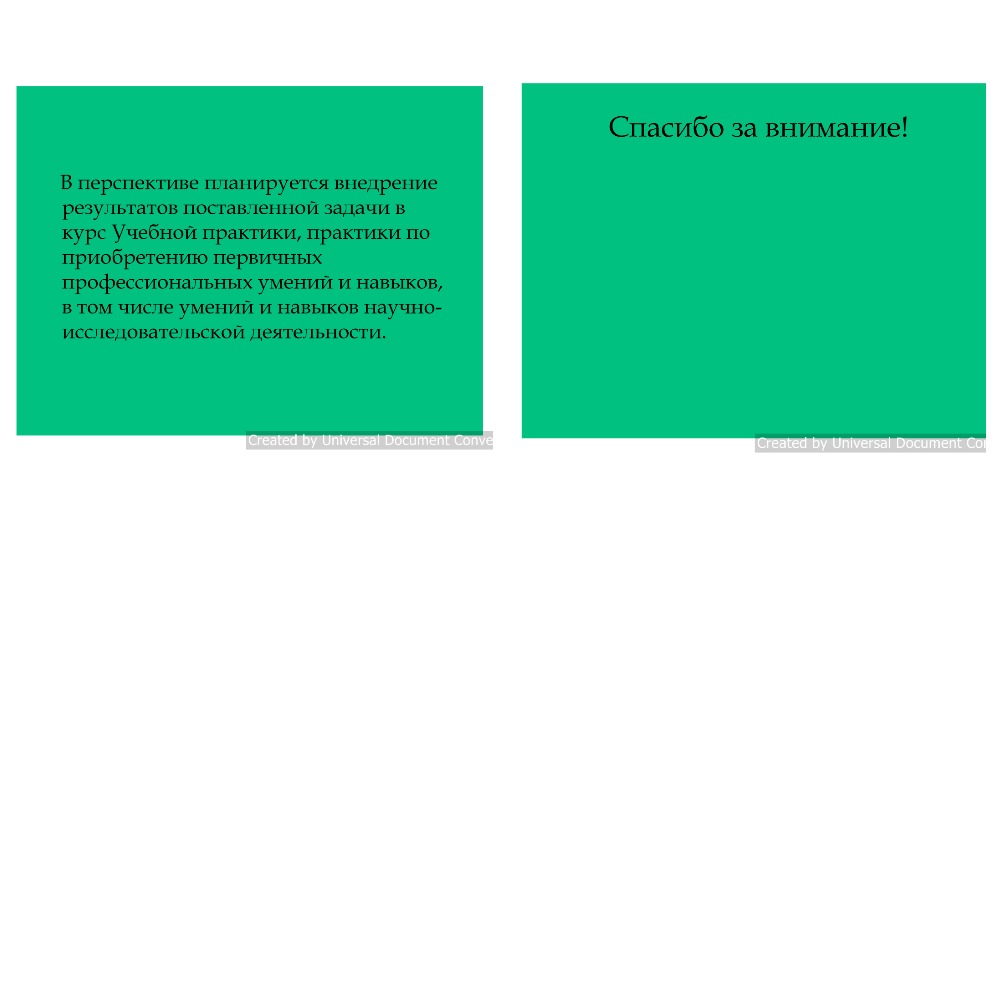
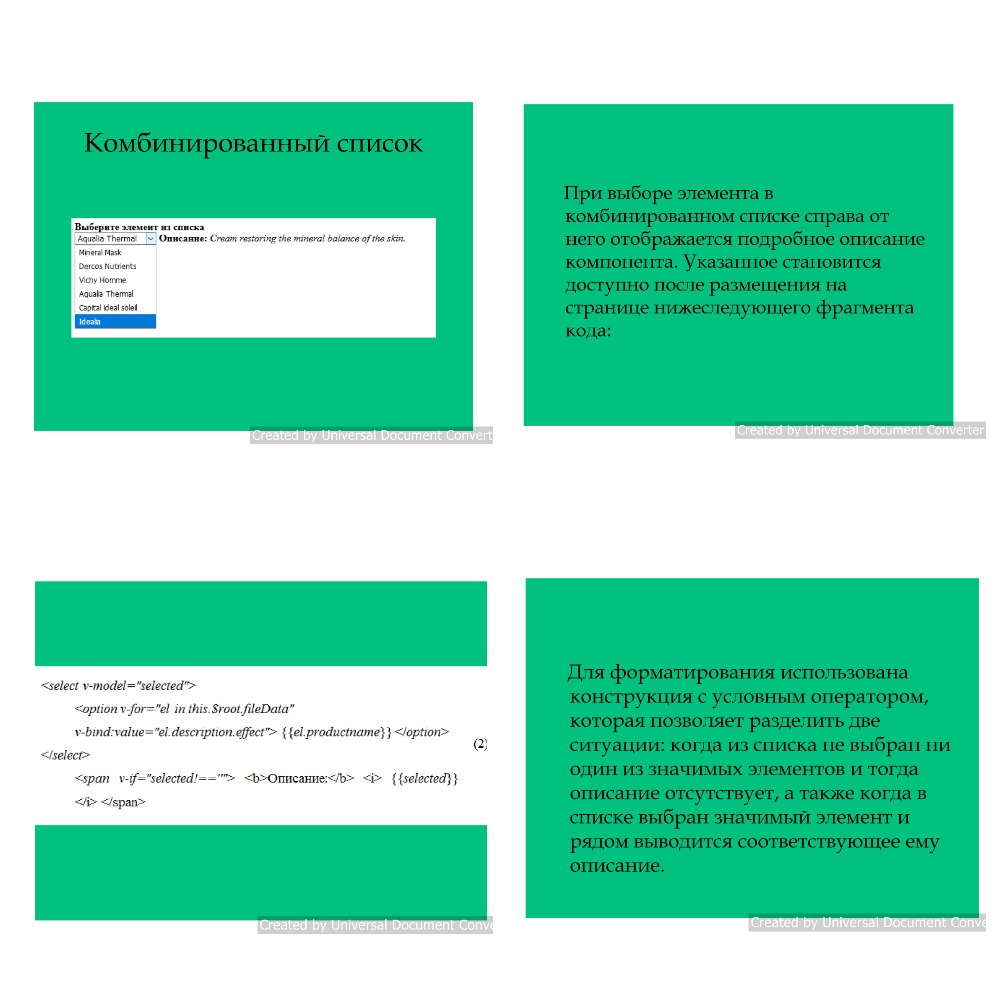
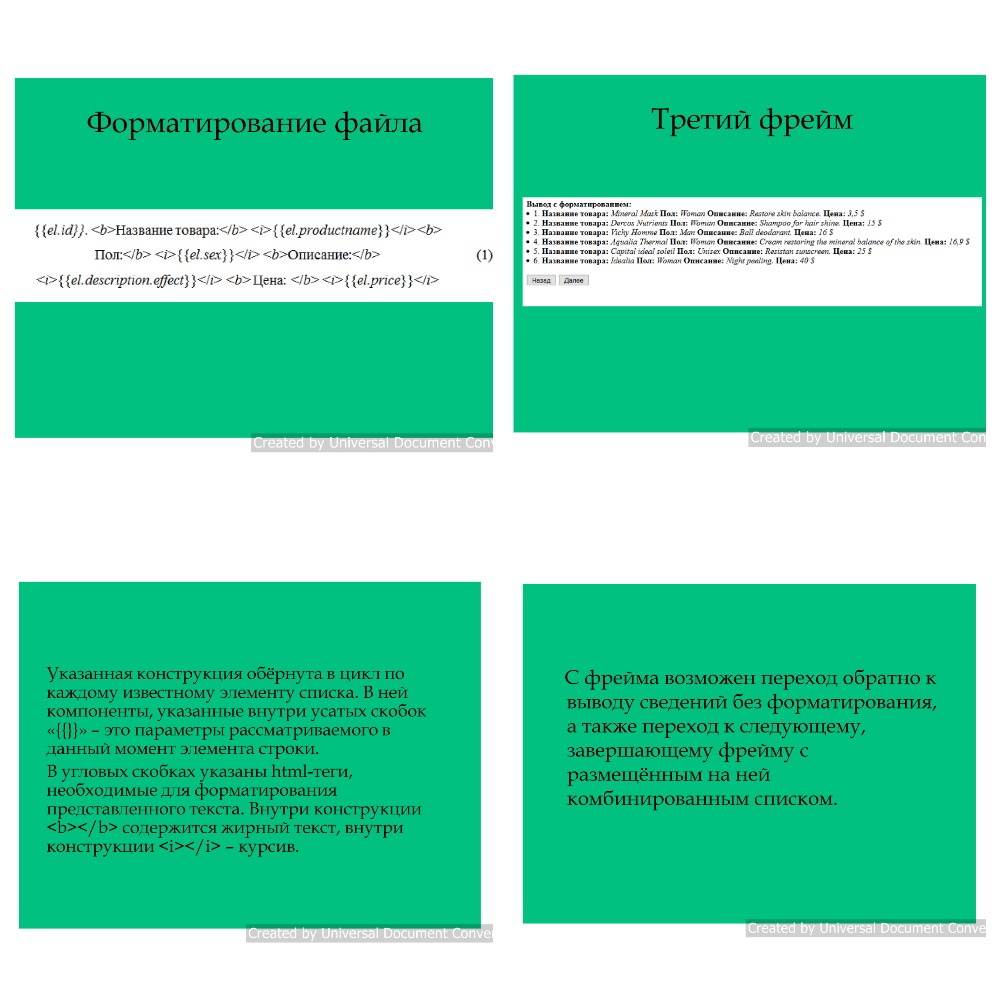
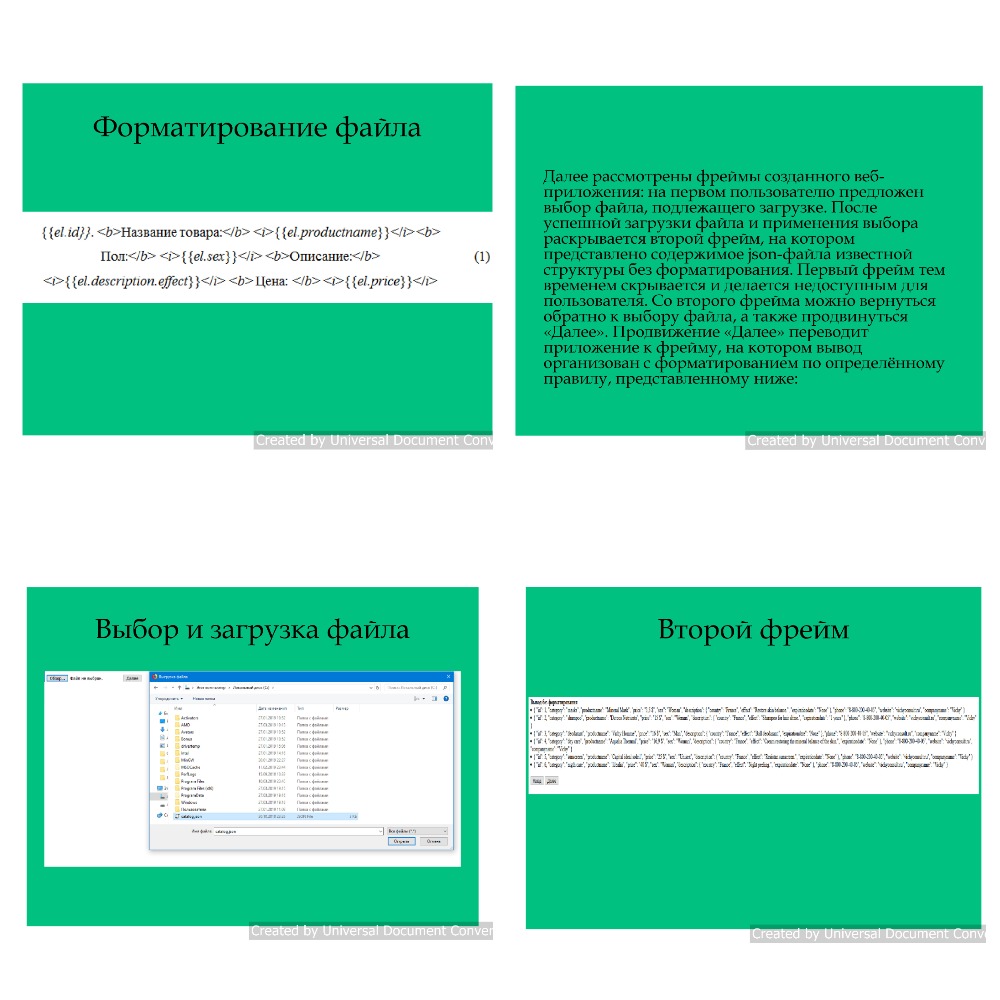
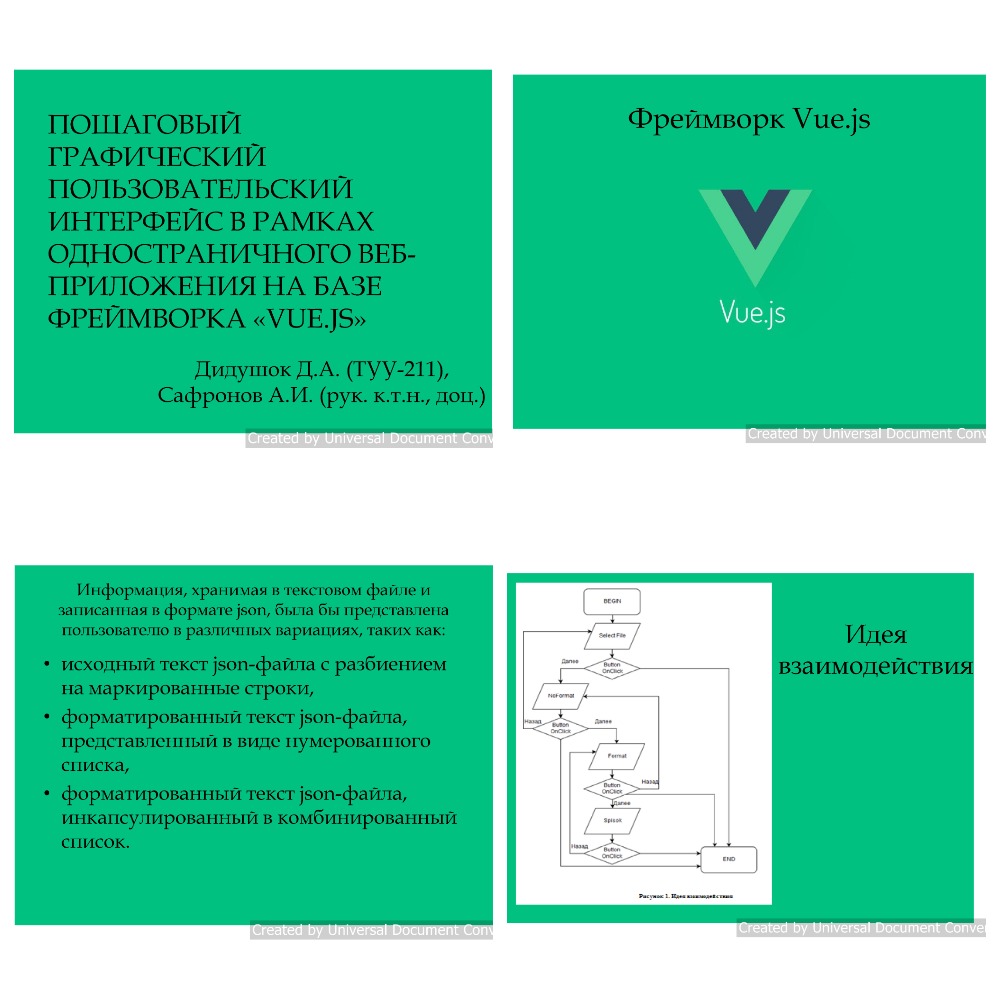
Для форматирования использована конструкция с условным оператором, которая позволяет разделить две ситуации: когда из списка не выбран ни один из значимых элементов и тогда описание отсутствует, а также когда в списке выбран значимый элемент и рядом выводится соответствующее ему описание.

В перспективе планируется внедрение результатов поставленной задачи в курс Учебной практики, практики по приобретению первичных профессиональных умений и навыков, в том числе умений и навыков научно-исследовательской деятельности.

Презентация 1:



Презентация 2:



Выводы:

• Создала браузерный калькулятора на базе фреймворка "vue.js" , который планирую применять и внедрять на кафедре УиЗИ.

• Разработала пошаговый пользовательский интерфес в рамках одностраничного веб-приложения на базе фреймворка "vue.js".

**Задание 2.**

Цель работы:

Изучать принцип работы с файлами с помощью JavaScript

**Задание:**

Разработать интерфейс, содержащий кнопки загрузки файлов, скачивания файла, изменения его расширения и добавление файлов в архив и скачивание его.

**Блок-схема:**

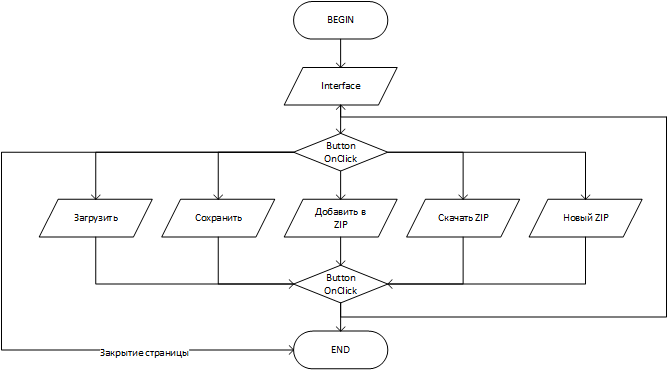
****

Рисунок 1. Идея взаимодействия

Выполнение работы:

* **Код:**

**HTML**

index.html

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<link rel="icon" href="image/favicon.ico" type="image/x-icon">

<link rel="stylesheet" href="style/main.css">

<title>Работа с файлами</title>

</head>

<body>

<script src="scripts/Vue.js"></script>

<script src="scripts/jszip.js"></script>

<script src="scripts/first\_page.js"></script>

<script type="text/x-template" id="firstPage">

<div id="vueFile">

<div id="app">

<divid="main">

Введите имя файла:

<div id="fileName" contenteditable></div>

Введитерасширениефайла:

<div id="fileExpansion" contenteditable></div>

<br>

<input id="fileInput" type="file" @change="readfile($event)" multiple>

<br>

<button id="select" @click="readerButton()">Загрузить</button>

<button id="download" @click="writefile()">Сохранить</button>

<p>

<button id="add" @click="addZip()">Добавить в Zip</button>

<button id="create" @click="createZip()">Скачать Zip</button>

<button id="clear" @click="clearZip()">Новый Zip</button></p>

</div>

<divid="result">Файлы в архиве:</div>

<div id="fileOutput" contenteditable>Введитетекст</div>

</div>

</div>

</script>

<div id="vueFile">

<component v-bind:is="getPage"></component>

</div>

<script>

varmyVueFileReader = new Vue(

{

el: '#vueFile',

data:

{

zip: '',

selPage: 'firstPage'

},

computed:

{

getPage: function()

{

zip = new JSZip();

if (this.selPage == 'firstPage')

{

return 'firstPage';

}

else

{

if (this.selPage == 'secondPage')

{

return 'secondPage';

}

else

{

if (this.selPage == 'direct')

{

return 'direct';

}

else

{

if (this.selPage == 'archiving')

{

return 'archiving';

}

}

}

}

}

}

})

</script>

</body>

</html> <divid="myCatalog">

<b>Выберите элемент из списка</b>

<br>

<select v-model="selected">

<option v-for="el in this.$root.fileData" v-bind:value="el.description.effect"> {{el.productname}} </option>

</select>

<span v-if="selected!==''"><b>Описание:</b><i>{{selected}}</i></span>

<br>

<button @click="back">Назад</button>

</div>

</script>

<div id="myCatalog">

<component v-bind:is="getPage"></component>

</div>

<script>

myVueCatalog = new Vue(

{

el: '#myCatalog',

data:

{

fileData: '',

pageN: 'fSelect'

},

computed:

{

getPage: function()

{

if (this.pageN == 'fSelect')

{

return 'selectf';

}

else

{

if (this.pageN == 'NForm')

{

return 'noform';

}

else

{

if (this.pageN == 'Form')

{

return 'format';

}

else

{

if (this.pageN == 'Spisok')

{

return 'nspis';

}

}

}

}

}

},

methods:

{

},

})

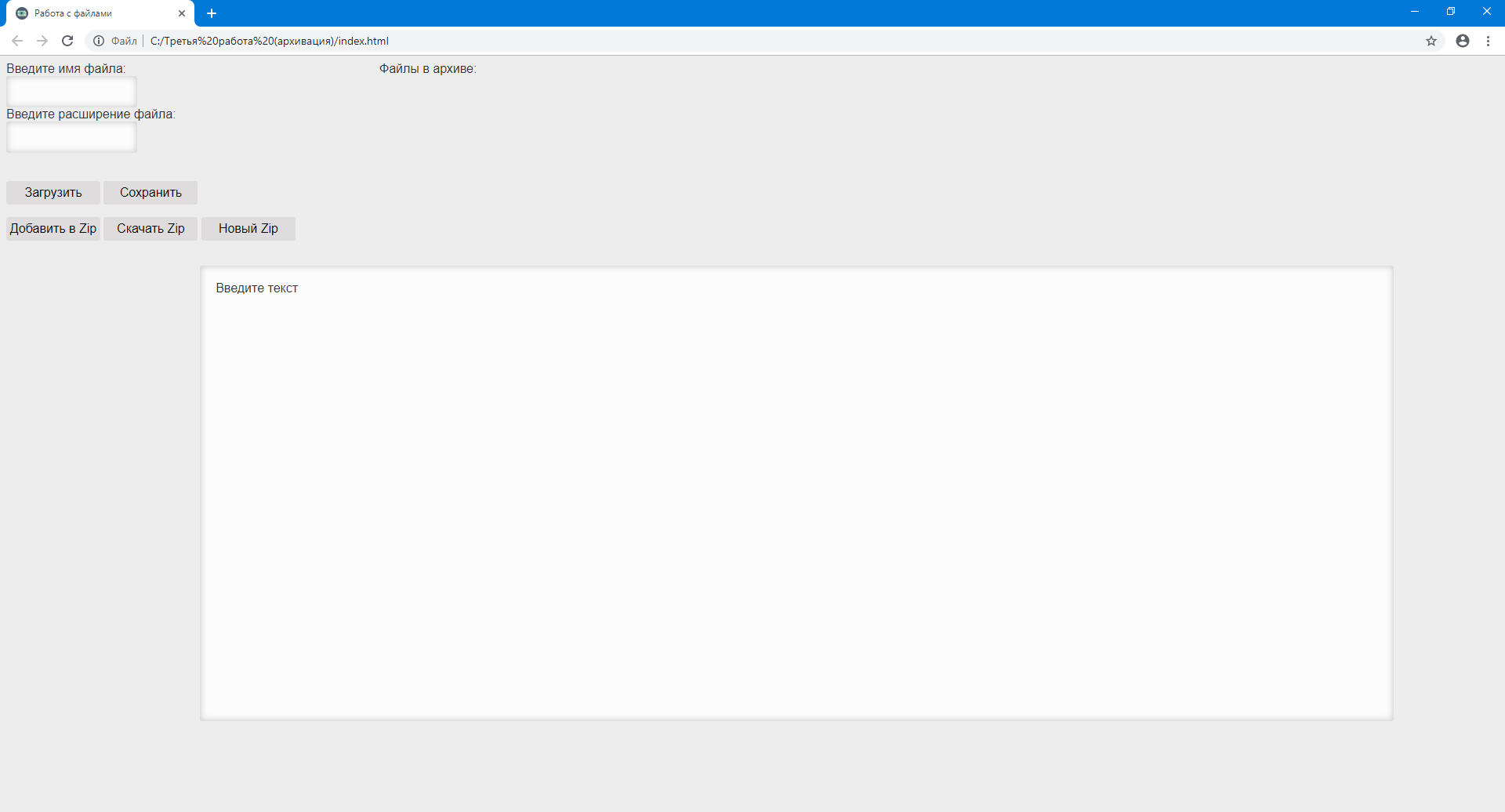
</script>

</body>

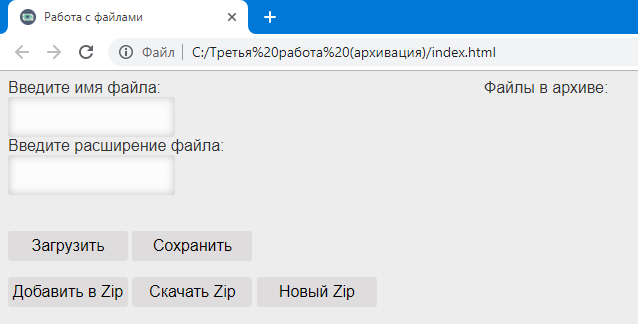
</html>

|  |
| --- |
| **JS** |
| first\_page.js |
| Vue.component('firstPage',  {  template: '#firstPage',  methods:  {  writefile: function ()  {  let content = document.getElementById('fileOutput').innerText;  let name = document.getElementById('fileName').innerText;  let expansion = document.getElementById('fileExpansion').innerText;  let blob = new Blob([content]);  let url = URL.createObjectURL(blob);  function download(url, name){  let a = document.createElement("a");  a.download = name;  a.href = url;  document.body.appendChild(a);  a.click();  a.remove();  }  download(url, name + '.' + expansion);  URL.revokeObjectURL(url);  },  readfile: function (e)  {  let file = e.target.ownerDocument.getElementById('fileInput').files[0];  tmp = file.name.split('.');  name = tmp[0];  expansion = tmp[1];  document.getElementById('fileName').innerText = name;  document.getElementById('fileExpansion').innerText = expansion;  let reader = new FileReader();  reader.onload = function (en) {  let output = document.getElementById("fileOutput");  output.textContent = en.target.result;  };  reader.readAsText(file);  },  readerButton: function()  {  let fileInput = document.getElementById('fileInput');  fileInput.click();  },  createZip: function()  {  zip.generateAsync({type:"base64"})  .then(function (content) {  location.href="data:application/zip;base64,"+content;  });  },  addZip: function()  {  let content = document.getElementById('fileOutput').innerText;  let name = document.getElementById('fileName').innerText;  let expansion = document.getElementById('fileExpansion').innerText;  zip.file(name + '.' + expansion, content);  let list = document.getElementById('result');  let block = document.createElement('div');  block.id = 'resultBlock';  if (document.getElementById('resultBlock') != null)  {  block = document.getElementById('resultBlock');  let li = document.createElement('li');  li.innerHTML = name + '.' + expansion;  block.appendChild(li);  }  else  {  list.appendChild(block);  block = document.getElementById('resultBlock');  let li = document.createElement('li');  li.innerHTML = name + '.' + expansion;  block.appendChild(li);  }  },  clearZip: function()  {  zip = new JSZip();  let list = document.getElementById('resultBlock').remove();  },  next: function()  {  myVueFileReader.selPage = 'direct';  }  }  }) |
| **CSS** |
| main.css |
| body {  font-family: Arial, Verdana, sans-serif; /\* Семействошрифтов \*/  font-size: 12pt; /\* Размер основного шрифта в пунктах \*/  background-color: #ededed; /\* Цвет фона веб-страницы \*/  color: #333; /\* Цвет основного текста \*/  }  button{  width: 120px;  height: 30px;  border-radius: 3px;  background: #dedcdc;  border: none;  box-sizing: border-box;  padding: 0;  font-size: 16px;  color: #000;  cursor: pointer;  outline: none;  }  button:hover {  border: 1px solid #000;  color: #000;  }  #fileOutput {  position: relative;  top: 260px;  right: 800px;  width: 80%;  height: 60vh;  margin: 0 auto;  padding: 20px;  background: #fcfcfc;  border-radius: 3px;  box-shadow: inset 0 0 8px 1px rgba(0,0,0,.2);  box-sizing: border-box;  overflow: hidden;  word-break: break-all;  outline: none;  resize: none;  flex: none;  }  #fileName {  position: relative;  width: 35%;  height: 0;  padding: 20px;  background: #fcfcfc;  border-radius: 3px;  box-shadow: inset 0 0 8px 1px rgba(0,0,0,.2);  box-sizing: border-box;  overflow: hidden;  word-break: break-all;  outline: none;  resize: none;  }  #fileExpansion {  position: relative;  width: 35%;  height: 0;  padding: 20px;  background: #fcfcfc;  border-radius: 3px;  box-shadow: inset 0 0 8px 1px rgba(0,0,0,.2);  box-sizing: border-box;  overflow: hidden;  word-break: break-all;  outline: none;  resize: none;  }  #fileInput {  display: none;  }  #app {  display: flex;  }  #main {  flex: none;  width: 25%;  }  #result {  flex: none;  width: 30%;  }  #next {  position: relative;  top: 30px;  }  #back {  position: relative;  top: 30px;  } |
|  |

**Результат:**

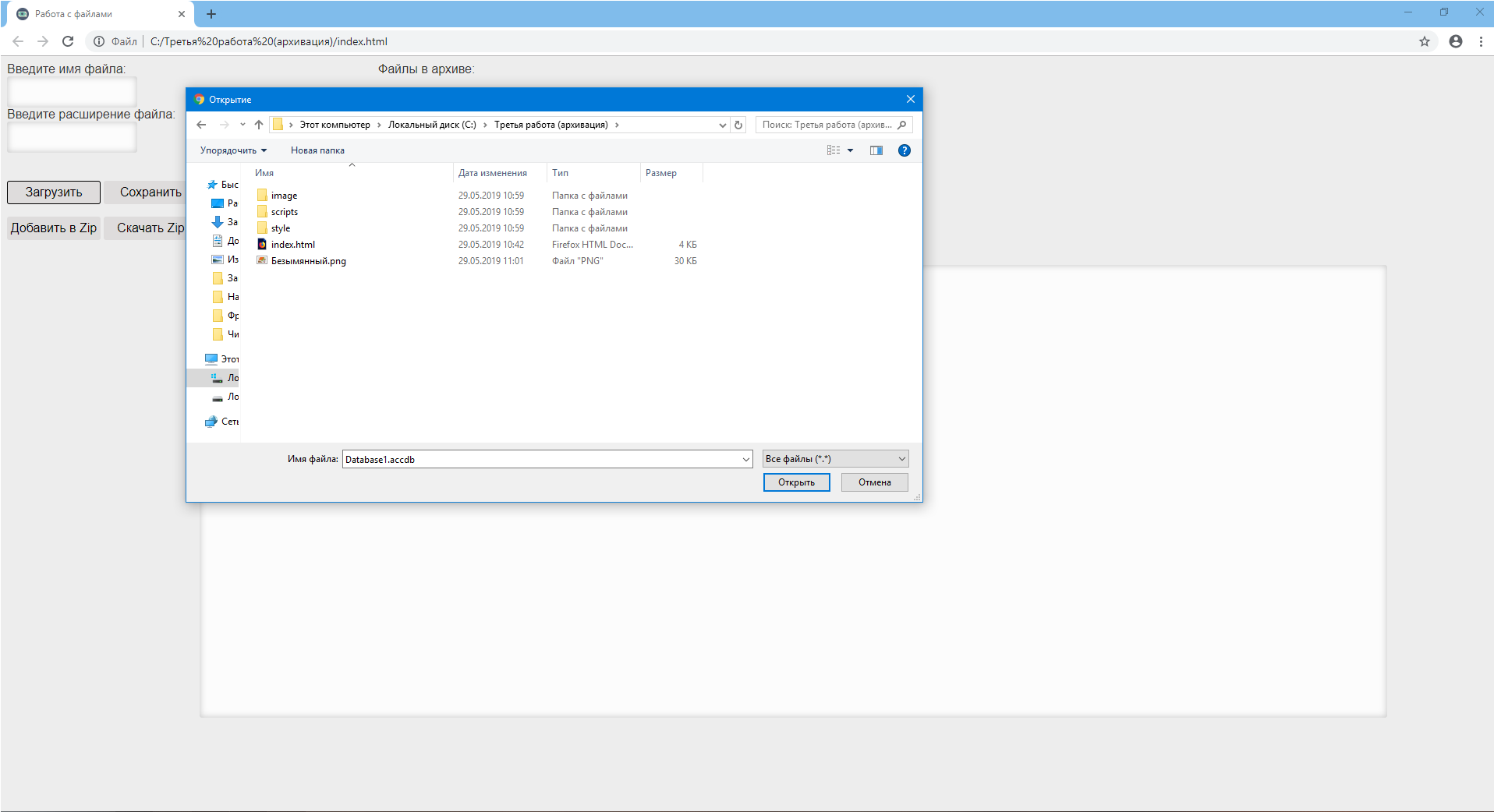
****

**Скриншот 1. Начальное окно**

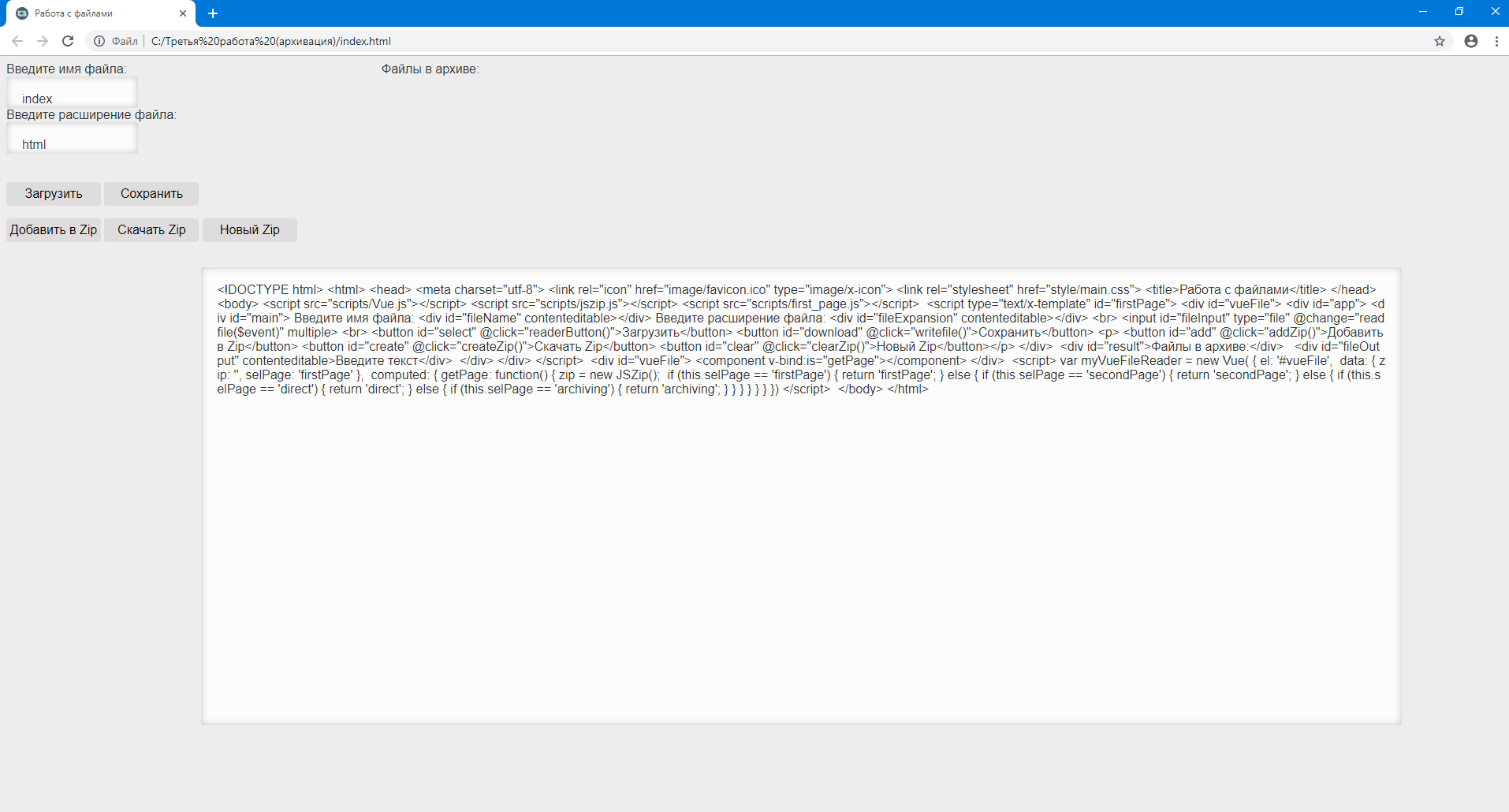


**Скриншот 2. Панель инструментов**

**При загрузке в окно "Введите имя файла" записывается его имя, "Введите расширение файла", его расширение. Возможность загрузки и сохранения файлов.**

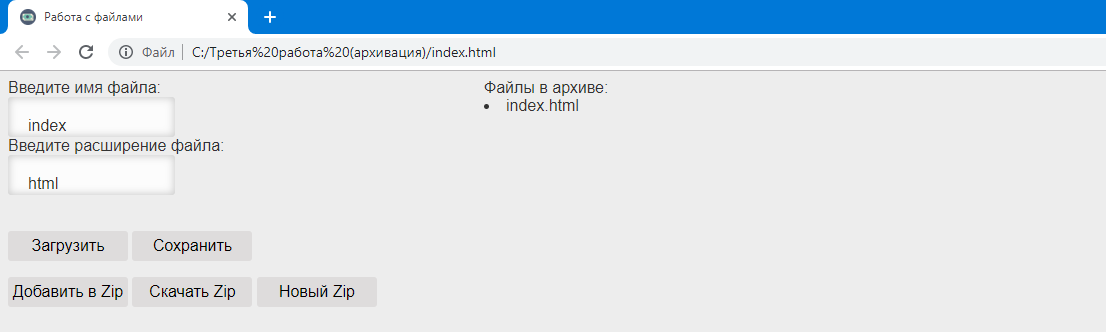
****

**Скриншот 3. Загрузка файла.  
С компьютера выбирается файл.**

****

Скриншот 4. Отображение данных из файла

**При загрузке файла имя и расширение файла вносятся в указанные поля, а все его содержимое- в общее поле.**



Скриншот 5. Отображение файлов в архиве

Вывод:

Изучены принципы работы c файлами с помощью JavaScript.