Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Факультет математики и информатики
Кафедра информационных и управляющих систем
Направление подготовки 09.03.02 – Информационные системы и технологии
Направленность (профиль) образовательной программы Безопасность информационных систем

ДОПУСТИТЬ К ЗАЩИТЕ
Зав. кафедрой
_____ А.В. Бушманов
«_____»_____2019 г.

БАКАЛАВРСКАЯ РАБОТА

на тему: Разработка веб-приложения по обработке изображений

Исполнитель
студент группы 555-об _____ А.В. Тищенко
(подпись, дата)

Руководитель
доцент, канд. физ.-мат. наук _____ В.В. Ерёмина
(подпись, дата)

Консультант
по безопасности и экологичности
доцент, канд. техн. наук _____ А.Б. Булгаков
(подпись, дата)

Нормоконтроль
инженер кафедры _____ В.Н. Адаменко
(подпись, дата)

Благовещенск 2019

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Факультет математики и информатики
Кафедра информационных и управляющих систем

УТВЕРЖДАЮ
Зав. кафедрой
_____А.В. Бушманов
«_____»_____ 2019 г.

З А Д А Н И Е

К бакалаврской работе студента Тищенко Андрея Владимировича

1. Тема бакалаврской работы: Разработка веб-приложения по обработке изображений
(утверждена приказом от 15.04.2019 №847-уч)

2. Срок сдачи студентом законченной работы _____

3. Исходные данные к бакалаврской работе: отчет о прохождении преддипломной практики, разработка проекта системы расчета и его реализация.

4. Содержание бакалаврской работы: анализ предметной области; проектирование информационной системы; разработка информационной системы; анализ угроз; рекомендации по обеспечению безопасности на предприятии.

5. Перечень материалов приложения: таблицы, программный код

6. Консультанты по бакалаврской работе: консультант по безопасности и экологичности, А.Б. Булгаков, доцент, канд. техн. наук

7. Дата выдачи задания: _____

Руководитель бакалаврской работы: Ерёмина Виктория Владимировна, доцент, канд. физ.-мат. наук

Задание принял к исполнению_____ А.В. Тищенко

РЕФЕРАТ

Бакалаврская работа содержит 72 с., 17 рисунков, 6 таблиц, 10 источников, 1 приложение.

ВЕБ-ПРИЛОЖЕНИЕ, ИНТЕРНЕТ, БАЗА ДАННЫХ, ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, ИЗОБРАЖЕНИЕ, ВЕБ-БРАУЗЕР, ИНТЕРФЕЙС, МОДУЛЬ ПРИЛОЖЕНИЯ, СЕРВЕРНЫЙ МОДУЛЬ, JAVASCRIPT, REACT, NODEJS, МОДЕЛЬ УГРОЗ, МОДЕЛЬ ЗЛОУМЫШЛЕННИКА

Веб-приложения – это программное обеспечение, работающее в окне веб-браузера и использующее возможности последнего для реализации схожих с настольными версиями функции прикладного программного обеспечения.

Целью работы является разработка веб-приложения по обработке изображений. Для достижения этой цели необходимо выполнить следующий перечень задач: анализ предметной области и разработка веб-приложения, включающий следующие функции:

- создание и авторизация пользователей;
- загрузка изображения;
- создание изображения;
- редактирование изображения;
- сохранения изображения.

На основе теории был разработано программное обеспечение, соответствующий поставленной задаче. Для создания веб-приложения была использована среда разработки Visual Studio Code, язык программирования JavaScript, библиотека для создания пользовательских интерфейсов ReactJS, серверная часть написана на NodeJS.

СОДЕРЖАНИЕ

НОРМАТИВНЫЕ ССЫЛКИ

В настоящей бакалаврской работе использованы ссылки на следующие стандарты и нормативные документы:

СТО СМК 4.2.3.21-2018 Оформление выпускных квалификационных и курсовых работ (проектов)

ГОСТ Р 51275-2006 Объект информатизации. Факторы, воздействующие на информацию

ГОСТ Р 53114-2008 Защита информации. Обеспечение информационной безопасности в организации. Основные термины и определения

ГОСТ 12.1.003-83 Система стандартов безопасности труда (ССБТ). Шум. Общие требования безопасности

СанПиН 2.2.2/2.4.1340-03 Гигиенические требования к персональным электронно-вычислительным машинам и организации работы

Федеральный закон от 21.12.1994 № 68-ФЗ О защите населения и территорий от чрезвычайных ситуаций природного и техногенного характера

Федеральный закон от 24.06.1998 №89-ФЗ Об отходах производства и потребления

СПИСОК ОБОЗНАЧЕНИЙ И СОКРАЩЕНИЙ

HTML – hypertext markup language – язык гипертекстовой разметки;

CSS – cascading style sheets – каскадные таблицы стилей;

JS – javascript;

SPA – single page application – одностраничные приложения;

SVG – scalable vector graphics – масштабируемая векторная графика;

PNG – portable network graphic – переносимая сетевая графика;

JPEG – joint photographic experts group;

RGB – red, green, blue – красный, зелёный, синий;

CMYK – cyan, magenta, yellow, black – голубой, пурпурный, жёлтый, чёрный;

HSV – hue, saturation, value – тон, насыщенность, значение;

HTTP – hypertext transfer protocol – протокол передачи гипертекста;

HTTPS – hypertext transfer protocol secure – безопасный протокол передачи гипертекста;

FTP – file transfer protocol;

IRC – internet relay chat;

JSON – javascript object notation;

DoS – denial of service – отказ в обслуживании;

DDoS – distributed denial of service – распределённая атака типа «отказ в обслуживании»;

TLS – transport layer security – протокол защиты транспортного уровня;

ПК – персональный компьютер;

ПЭВМ – персональная электронно-вычислительная машина;

ПО – программное обеспечение;

ОС – операционная система;

ВДТ – видеодисплейный терминал;

ПДК – предельно-допустимая концентрация.

ВВЕДЕНИЕ

Развитие интернет-технологий и устройств с подключением к сети Интернет идёт очень высокими темпами. Жизнь человека уже сложно ввести продуктивно и эффективно без Всемирной паутины. На 2019 год число уникальных пользователей в сети составило 4,388 млрд человек, из которых 3,196 миллиарда считаются активным пользователя социальных сетей [6]. Во многом это происходит благодаря развитию и распространению мобильных сетей таких, как 3G и 4G, а также всё возрастающей потребности человека в наличии интернета для поддержания социальных связей.

В связи с таким развитием Интернета появилась нужда в разработке веб-сайтов. В связи с этим с каждым годом появляются множество платформ, онлайн-сервисов, веб-сайтов и веб-приложений. Благодаря наличию развивающихся интернет-стандартов позволяет веб-разработчиком дополнять и улучшать как функционал их приложений, так и взаимодействие программ. Каждый день пользователь посещает различные ресурсы для удовлетворения потребности в актуальной информации, а также социальные сети – для общения, обмена изображениями или просмотра видео.

С развитием таких веб-технологий, как HTML, CSS и JS веб-сайты становятся динамическими, более дружелюбными к пользователю и имеющими более привлекательный пользовательский интерфейс с плавной анимацией. Например, вместо выдачи сервером статической страницы, которая для любого перехода заставляет перезагружать всю страницу, теперь сервисы загружают лишь ту часть, что требуется для пользователя в данный момент. Помимо этого, появляются и так называемые SPA-приложения, которые работают на стороне клиента, необходимые файлы скачиваются при запросе к серверу.

Также развивается и серверная часть, чтобы соответствовать новым нормам при разработке. Сайты разрабатываются преимущественно на архитектуре «клиент – сервер», которая представляет собой систему взаимосвязанных модулей.

Серверный модуль разрабатывается для работы с пользовательским интерфейсом, который принимает запросы от пользователей. Одним из частых запросов является авторизация пользователей. Несмотря на то, что авторизация имеется на многих сайтах и имеет схожие черты: электронная почта, пароль и имя пользователя для регистрации или входа – их работа на стороне сервера может отличаться: это зависит от того, какие данные пользователя будут собраны и защищены.

Целью данной бакалаврской работы является разработка веб-приложения для обработки изображений.

Задачами бакалаврской работы, в связи с указанной целью, являются:

- анализ предметной области;
- проектирование информационной системы;
- разработка информационной системы;
- исследование угроз информационной безопасности;
- разработка рекомендаций по обеспечению безопасности при работе с программным продуктом.

Создание данного программного обеспечения позволит обеспечить быстрое создание, редактирование и сохранение изображений для дальнейших манипуляций пользователя.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Теория компьютерной графики

Компьютерная графика – это одна из областей информатики, занимающаяся изучением создания, хранения и обработки изображений на вычислительных устройствах [1, с. 9].

Компьютерная графика зародилась в начале 1950-х годов и с тех пор развивается по мере совершенствования компьютеров и их внедрения в нашу жизнь. Увеличились объёмы памяти, произошли качественные скачки как в обработке, так и в отображении графики – разрешение, яркость, контрастность и цветопередачи дисплеев улучшились, а для их обработки появились специализированные устройства, которые получили название графический процессор (GPU). С недавних пор данная тенденция охватывает и мобильные устройства, которые укрепляют позиции в жизни человека и зачастую используются чаще самих персональных компьютеров. Их способность и обрабатывать, и хранить визуальную информацию постоянно развивается как с программной, так и аппаратной точки зрения.

С технической стороны графику принято делить на **трёхмерную** – в которой операции с изображением происходят в виде объектов в пространстве и используется пакеты моделирования 3D объектов, а также **двухмерную** – где описание изображения происходит в виде двух координат, используются специализированные программные продукты для создания и редактирования визуальной информации, именуемые графическими редакторами.

В рамках данной выпускной работы будет затронута именно работа с двухмерными изображениями.

В теории компьютерной графики принято выделять три типа представления и хранения двухмерных изображений: растровый, векторный и фрактальный.

При **векторном способе** визуальная информация представляется в виде множества геометрических примитивов – кривых, дуг, отрезков и т.п. Главным

достоинством такого метода является высокая точность воспроизведения информации. Это связано с хранением её в аппаратно-независимых единицах (от англ. device-independent pixels). Такой способ отображения и хранения графической информации идеален для чертежей, логотипов, иконок, где не требуется фотореализм. Векторные графические редакторы позволяют проводить операции перемещения, отражения, растягивания, скашивания, основные аффинные преобразования, менять порядок и комбинировать примитивы в более сложные. Примеры таких редакторов – Inkscape, Illustrator от Adobe, CorelDraw от Corel и множество других. Популярными форматами хранения векторной графики являются SVG, AI, CDR и другие.

Ещё один популярный способ хранения визуальной информации – **растровый**, его принцип хранения и отображения таков: на экран монитора выводится матрица (MxN) элементов, где каждый элемент – пиксел (от англ. picture element), он может принимать значения в диапазоне цветового множества или цветового пространства. Среди главных преимуществ такого способа выделяют естественность вывода, что позволяет достичь высокой скорости обработки изображений (чего, однако, нельзя сказать об их масштабировании), а также возможность создавать графической информации любой сложности, ввиду простоты передачи переходов цветов. Ввиду популярности данного способа существуют множество различных редакторов, позволяющих проводить как самые базовые операции над изображениями – масштабирование, раскраску, обрезание, рисование, так и самые продвинутые – послойную работу с изображением, настройку кривой яркости, различные эффекты обработки. Примерами таких редакторов являются Paint от Microsoft, Photoshop и Lightroom от Adobe, Pixelmator Pro от Pixelmator Team и т.д. Так как растровое изображение представляют собой матрицу, то хранение данного вида информации затрудняется ввиду объёмов, которые она занимает на накопителях. Это привело к появлению алгоритмов сжатия без потерь – PNG, Lossless JPEG, TIFF и с потерями – JPEG, WebP.

Фрактальный способ хранения и отображения основывается на фрактале – объекте, элементы которого наследуют свойства структур родителей. Их

проще математически описать и используются исключительно для детального описания классов изображений, но описание возможно только для ограниченного множества вещей, что и является ограничением данного вида графики.

Стоит отметить, что компьютеры могут использовать для отображения только пиксели, поэтому в случае с векторной или фрактальной видами для отображения изображение растеризовают или, иначе говоря, представляют в виде матрицы с пикселями. На современных графических процессорах это реализовано аппаратно.

При работе с визуальной информацией невозможно не затронуть такое базовое понятие как цвет. Для описания цвета прибегают к математической модели, чтоб описать цвета в виде кортежей чисел, которые называют цветовыми компонентами, а саму модель именуют как цветовую. В теории компьютерной графики существует несколько таких представлений: RGB, CIE XYZ, CMYK и HSV.

RGB – это цветовая модель со свойством аддитивности, которая представляют любой цвет в виде суммы трех цветов: Red (красный), Green (зелёный) и Blue (синий). Выбор таких цветов обусловлен физиологическими особенностями восприятия цвета глаза, однако на практике оказывается невозможно описать все цвета или чистые спектральные цвета.

$$\begin{pmatrix} r \\ g \\ b \end{pmatrix} = \begin{pmatrix} 0-255 \\ 0-255 \\ 0-255 \end{pmatrix} \quad (1)$$

где r – красный;

g – зелёный;

b – синий.

Поэтому была предложена более эталонная модель – CIE XYZ, ставшая стандартом представления цвета, а также выражения других цветовых моделей. Несмотря на это, модель RGB используются чаще всего ввиду её простоты и достаточности.

Для получения XYZ-координат цвета по спектральной функции используют следующие формулы:

$$X = \int I(\lambda)k_x; Y = \int I(\lambda)k_y; Z = \int I(\lambda)k_z \quad (3)$$

где x – координата цвета по оси X;

y – координата цвета по оси Y;

z – координата цвета по оси Z.

CMYK – субтрактивная цветовая модель, которая пришла из печати из-за иного подхода к воспроизведению цвета: при цветной печати на поверхность бумаги наносятся слои красителей, отражающие определённые цвета и поглощающие другие, поэтому принято было рассматривать нанесение красителя как операцию вычитания цвета из белого (цвет листа бумаги). Используемые же «опорные» цвета в модели – это Cyan (голубой), Magenta (пурпурный), Yellow (жёлтый) и Black (черный).

Для перевода между моделями RGB и CMY используют следующую формулу:

$$\begin{pmatrix} r \\ g \\ b \end{pmatrix} + \begin{pmatrix} c \\ m \\ y \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \quad (3)$$

где r – красный;

g – зелёный;

b – синий;

c – голубой;

m – пурпурный

y – жёлтый.

HSV – цветовая модель, пользующаяся популярностью у художников и фотографов. И цвет представляется другим способом через Hue (тон), Saturation (насыщенность) и Value (яркости). Насыщенность показывает, насколько цвет

близок к серому той же яркости и принимает значения от 0 (серый цвет) до 1 (чистый цвет), а тон является оттенком чистого цвета и представляется как угол, изменяющийся от 0 до 360 градусов. Яркость принимает такие же значения, как и насыщенность. Следовательно, множество допустимых цветов является перевёрнутой шестигранной пирамидой, где вершина – это чёрный цвет, основания – это цвета с наибольшей яркостью, боковые грани – чистые цвета различной яркости и высота – оттенки серого цвета.

Для преобразования из RGB в HSV используют следующие формулы:

$$\begin{cases} V = M \\ S = \frac{d}{M} \\ H = \begin{cases} \frac{g-b}{d}, M = r \\ \frac{b-r}{d}, M = g \\ \frac{r-g}{d}, M = b \end{cases} \end{cases} \quad (4)$$

где $M = \max(r, g, b)$;

$m = \min(r, g, b)$;

$d = M - m$.

1.2 Современный Web – веб-приложения и веб-сервисы

В настоящее время Всемирная Паутина считается платформой, на основе которой разработаются различные приложения и сложные информационные системы. В качестве архитектур систем или приложений разработчики используют «клиент-сервер», где клиентом выступают пользователи с web-браузером, а сервером – web-сервером (HTTP-серверы), принимающие HTTP-запросы от клиентов на получение различного рода ресурсы – HTML-страницы, графические файлы, медиа-потоки и т.п. В ответ сервер передаёт HTTP-ответ, включающий запрошенные данные [2, с. 61].

Web-браузер – это программное обеспечение на стороне клиента, позволяющее пользователям выполнять запросы к web-серверам по протоколу HTTP, получать ответы и выполняет следующие операции:

- поддержка взаимодействия с различными протоколами – FTP, HTTP, HTTPS, IRC;

- отображение содержания различных типов ресурсов (от HTML-документов до медиа-файлов);

- выполнение переходов по ссылкам, исполнение скриптов на страницах;

- ввод данных пользователя с помощью форм и передачи их серверу с помощью методов GET или POST;

- кэширование – временное хранилище ресурсов, получаемых от web-серверов;

- аутентификация и авторизация через специальную формы;

- поддержка состояний через cookie, который содержит информацию, необходимую серверу для последующих запросов;

- запрос поддерживающих элементов данных для правильного отражения или работы ресурсов на странице;

- обработка ошибочных состояний – ошибок соединения, неправильных ответов серверов и других аналогичных ситуаций.

Стоить отметить, что распространяются браузеры, как правило, бесплатно, что положительно сказывается на их популярности. Самыми популярными браузерами на октябрь 2018 считаются Google Chrome с долей в 69,84 %, Mozilla Firefox – 10,14 %, Internet Explorer – 6,01 %, Safari – 5,61 % и Edge – 4,21 % [10].

Web-сервер – это программа, которая работает в фоновом режиме и ожидает запросы для дальнейшего их обработки. Он принимает HTTP-запросы от клиентов (обычно web-браузеры) и возвращает HTTP-ответы. К функциям относят:

- виртуальный хостинг, который позволяет на одном web-сервере хранить под различными именами и, соответственно, адресами web-сайты;

- разрешение адреса – определение типа запрашиваемого контента на основе URL-пути в реальный адрес файловой системе сервера;

- аутентификация – проверка данных авторизации, чтобы определить имеет ли право пользователь использовать данный ресурс;

- передача данных поступающих сообщений и параметров сервера приложению и возврат сформированного кода серверу;

- журнализация обращений пользователей к ресурсам;

- поддержка HTTPS для защищённости соединения.

Ввиду популярности Всемирной Паутины в различных сферах разрабатываются веб-приложения, под этим подразумевается «клиент-сервер» программа, которая использует web-браузер и работает на web-сервере. Как правило, при таком сценарии сначала запускается программа, формирующая динамическую HTML-страницу, которая затем возвращается её клиенту; основная логика работы лежит на стороне web-сервера. В зависимости от используемых технологий web-приложение может быть не самостоятельной программой, а выполняться в среде выполнения – внешней программе. В качестве таких программ выступают: сервер-приложений или контейнер приложений. Среды выполнения выполняют следующие задачи:

- приём от web-сервера всей информации по запросу;

- определение выполняемого приложения;

- создание необходимых для работы объектов (Application, Session, Request, Response, User и т.п.);

- загрузка и передача управления запрашиваемому приложению;

- создание приложением HTML-страницы и её запись, как правило, в объект Response;

- формирование Web-сервером HTTP-ответа, включая объект(-ы) от приложения), и отправка их клиенту.

В среде web-приложений существует такое понятие как сеанс, определяющееся как последовательность вызовов web-приложения от пользователя. Сеанс завершается в результате выбора команды об окончании или длительного промежутка времени между запросами.

Среди главных преимуществ web-приложений выделяют:

- простоту развёртывания или установки, так как не требуется установка на компьютерах пользователей, достаточно перейти по URL-адресу приложения.

При любых изменениях или обновлениях клиенты сразу работают с новой версией;

- простота доступа к приложению – любой человек с компьютером и доступом к сети Интернет может использовать приложение;

- высокий уровень развития и надёжности web-технологий и сетей.

Недостатками же являются:

- слабосвязанная архитектура – отсутствует поддержка состояния сеанса и задержки при перезагрузке страницы;

- открытость доступа к приложениям: так как есть возможность получить доступ к исходному коду из web-браузера, требуется поддерживать безопасность работы пользователей с ними;

- ограниченный набор элементов пользовательского интерфейса для форм приложения, так как HTML не предполагает сложные взаимодействия.

Помимо web-приложений выделяют еще и web-сервисы, которые представляют из себя набор методов или программный интерфейс для взаимодействия через HTTP-запросы, которые выдают результат в виде HTTP-ответов. Это все означает, что они могут выступать компонентами более сложных систем – как часть web-приложения или web-сервера, однако они не предназначены для работы с web-браузерами, потому что не имеют графического интерфейса и не служат для прямого взаимодействия с клиентом.

Технические web-сервисы реализуются на тех же web-технологиях, что и web-приложения, и имеют свой URL-адрес для обращений. Для работы и выдачи результатов могут использоваться XML, JSON или YAML, чаще используется JSON ввиду популярности.

1.3 Сравнение графических редакторов

Microsoft Paint – растровый графический редактор от компании Microsoft, входящий в состав операционной системы Windows. Данный редактор поддерживает следующие инструменты: заливка, кисть, пипетка, увеличение, вставку, выделение, обрезание, повороты.

Из плюсов можно выделить – относительную многофункциональность и простоту использования.

Недостатками же является: доступность только на ОС Windows, отсутствие слоёв, невозможность указать размер при создании файла, повороты, кратные только 90 градусам, отсутствие прозрачности и устаревший интерфейс.

Adobe Photoshop – гибридный графический редактор от Adobe Systems. В настоящее время широко применяется в графическом дизайне. К его плюсам можно отнести: широкий функционал работы над изображениями, поддержка различных эффектов, большой выбор цветовых моделей, поддержка слоёв, поддержка прозрачности, поддержка Windows и macOS.

Минусами являются: «примитивная» поддержку векторной графики, платная лицензия, закрытый исходный код, мало функциональные мобильные версии приложения, а также отсутствие официальной версии приложения под Linux.

GIMP – гибридный графический редактор, который свободно распространяется и поддерживается Gnome Foundation и другими разработчиками. Функционально богат и очень похож на Adobe Photoshop, хоть и с небольшими ограничениями – отсутствие поддержки мобильных платформ в любом виде также, как и поддержки цветовых пространств CIELAB и CIEXYZ.

Adobe Illustrator – это векторный графический редактор от Adobe Systems, широкое применение получил в веб-дизайне и дизайне при разработке логотипов и реклам. Значительно ограничен в отличие от GIMP или Photoshop, хоть и имеет намного богатый функционал по работе с векторной графикой – создание и экспорт в SVG, рисование кривых и пользовательских фигур, создание масок, обрезание, выделение, объединение.

Pixlr Photo Editor – это растровый графический редактор от Pixlr, сделанный в качестве веб-приложения с минималистичным дизайном и выбором двух тем – светлой или тёмной. Функционально небогат и предоставляет небольшой набор для редактирования только существующих изображений, так как функция создания отсутствует. Помимо этого, программа поддерживает только английский язык.

miniPaint от разработчика ViliusL является растровым редактором изображений, повторяющий функционал Microsoft Paint с некоторыми дополнениями: слои, предпросмотр, детали слоя, различные эффекты и поиск картинок. К сожалению, из поддерживаемых автором языков является только английский.

В ходе сравнения графических редакторов мы пришли к выводу об отсутствии на рынке приложения, доступного для любого русскоговорящего человека с web-браузером и подключением к сети Интернет, которое бы удовлетворяло как простых пользователей и их потребность в удобном обмене из приложения результатов работы. Из чего возникает практическая необходимость в создании подобного рода продукта.

При этом приложению необходима поддержка как мобильных, так и настольных версий web-браузеров, чего можно достичь при использовании современных web-технологий. Также целесообразно предоставлять исходный код для того, чтобы продвинутые пользователи могли по своему желанию изменять и улучшать функционал приложения.

На основе этого будет разработан web-приложение для обработки изображений, который облегчит работу с графической информацией, предложит пользователю удобный графический интерфейс и будет постоянно доступен по URL-адресу.

2 ПРОЕКТИРОВАНИЕ ВЕБ-ПРИЛОЖЕНИЯ

Проектируемая программа состоит из двух частей – серверной и клиентской. Серверная часть программы должна будет выдавать информацию из локальной базы данных клиентской части, а последняя должна предоставлять полноценный интерфейс для работы. Структурная схема работы программного продукта показана на рисунке 1.

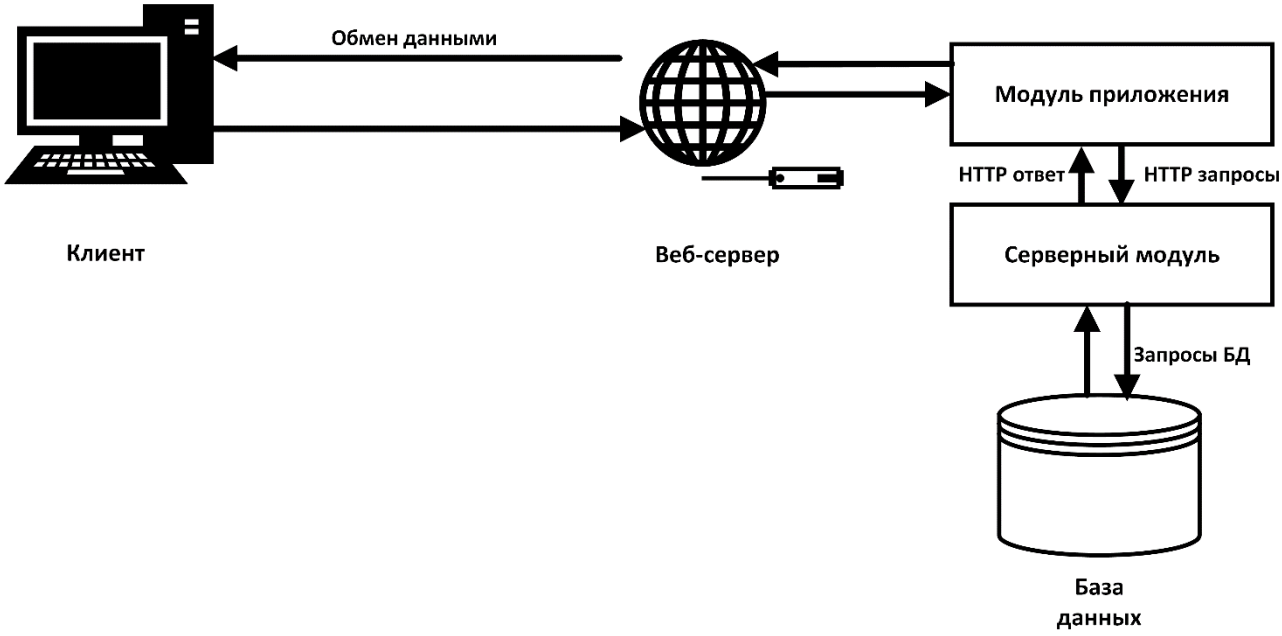

Рисунок 1 – Структурная схема работы веб-приложения

Для работы с изображениями, в системе необходимо авторизоваться. После успешной авторизации предоставить интерфейс приложения, а при неудачной – сообщить об ошибке и\или предложить зарегистрироваться в информационной системе. При работе с приложением у пользователя будет представлен ряд инструментов для обработки с изображениями, а также следующие возможности: сохранить результат на компьютер пользователя и создать новое изображение. Функциональная модель программы представлена на рисунке 2, а её декомпозиция на рисунке 3.

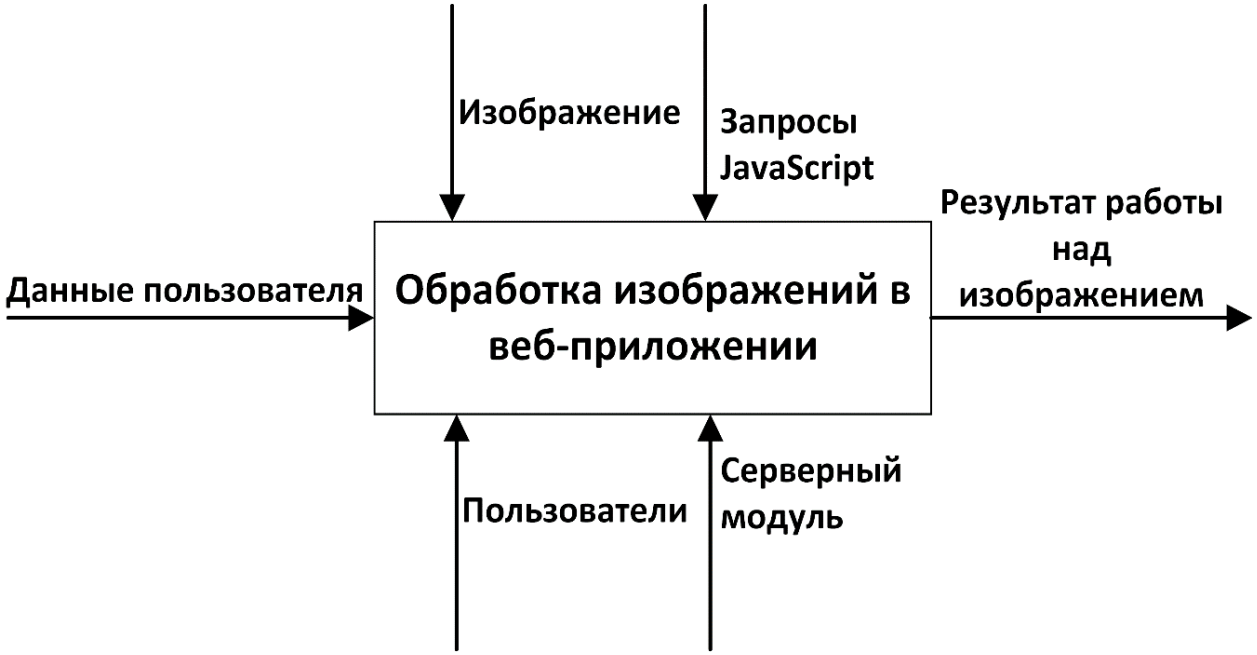

Рисунок 2 – Функциональная модель программы в нотации IDEF0

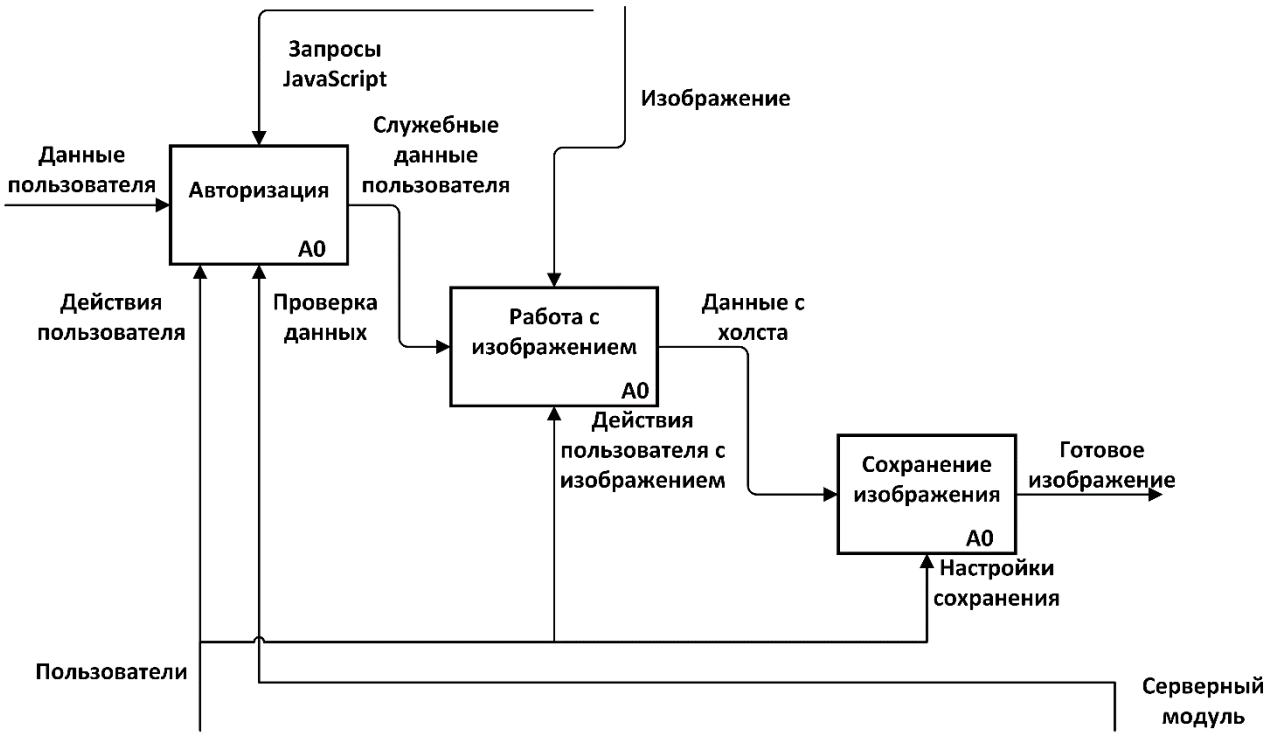

Рисунок 3 – Декомпозиция функциональной модели программы

2.1 Цели и назначение программы

Программа предназначена для работы с графической информации, которая может создаваться как с нуля, так и на базе существующей при помощи загрузки изображения.

Цели создания веб-приложения – это работа с изображениями в веб-браузере для пользователей сети Интернет, которые могут посетить ресурс и обработать свою картинку, так и создать новую.

Основные задачи программы:

- регистрация и авторизация пользователей, а также выход их из системы;
- создание нового изображения;
- выбор инструмента для работы с изображением;
- редактирование изображением выбранным инструментом;
- загрузка изображения;
- сохранение изображения на накопитель ПК пользователя.

2.2 Характеристика функциональных модулей программы

Главные функции программы:

- функция «создание пользователя» обеспечивает регистрацию пользователя, если его нет в системе, для дальнейшей работы с приложением;
- функция «авторизация пользователя» отвечает за соответствие введенных данных пользователя с хранимыми в базе данных серверной части. В случае соответствия, приложение одобряет вход в систему;
- функция «сброса пароля» позволяет пользователям, которые утратили доступ к своей учетной записи, восстановить пароль, если помнят электронный адрес, который указывали в регистрации;
- функция «боковая панель инструментов» предлагает авторизованным пользователям такие инструменты, как: добавление шрифтовых символов из библиотеки, добавления текста, изображения, векторных фигур – треугольник, прямоугольник, круг, линий, стрелок полигональных линий; кисточки и заливки;
- функция «заголовочная панель управления» предназначена для управления добавленных элементов с боковой панели инструментов, размещения элементов на холсте, создания и сохранения результатов работы;
- функция «боковая панель инструмента и холста» предоставляет более тонкую настройку выбранного инструмента с боковой панели: цвета, толщины, тени, размера холста;

- функция «рабочая область» объединяет результаты работ, которые сделаны пользователями, и отображает их на холсте;

- функция «выхода пользователя» обеспечивает выход пользователя из системы с занесением соответствующей записи в историю пользователя, которая хранится в базе данных;

- функция «смена темы» обеспечивает перемену темы со светлой на темную и наоборот, чтобы снизить нагрузку на глаза при работе в тёмном помещении.

На основе этих функций можно выделить модули:

- модуль приложения – программный код, который реализует непосредственную работу пользователями с графической информацией;

- серверный модуль – программный код, который реализует создание, сохранение, выдачу данных учетных записей по запросу от модуля приложения.

Эти модули необходимы для корректной работы программы. Далее стоит определить этапы работы модулей.

Этапы работы серверного модуля:

- этап 1 – ожидание поступающих запросов от модуля приложения. В случае запрос на добавление нового пользователя, перейти к этапу 2. В случае авторизации текущего пользователя, перейти к этапу 3. В случае выхода пользователем системы, перейти к этапу 4;

- этап 2 – проверка полученных данных с запроса на добавление. В случае успешной проверки, перейти к этапу 5. В случае неуспешной проверки (например, пользователь существует в системе) переход к этапу 6;

- этап 3 – проверка полученных данных с запроса для авторизации. В случае успешной проверки, перейти к этапу 5. В случае неуспешной проверки (например, пользователь неправильно ввел пароль), перейти к этапу 6;

- этап 4 – обновить запись учетной записи с добавлением к истории пользователя о состоянии выхода из системы;

- этап 5 – передать данные учетной записи модулю приложения. Перейти к этапу 1;

- этап 6 – передать ошибку выдачи данных пользователя модулю приложения. Перейти к этапу 1;

Диаграмма данного модуля представлена на рисунке 4.

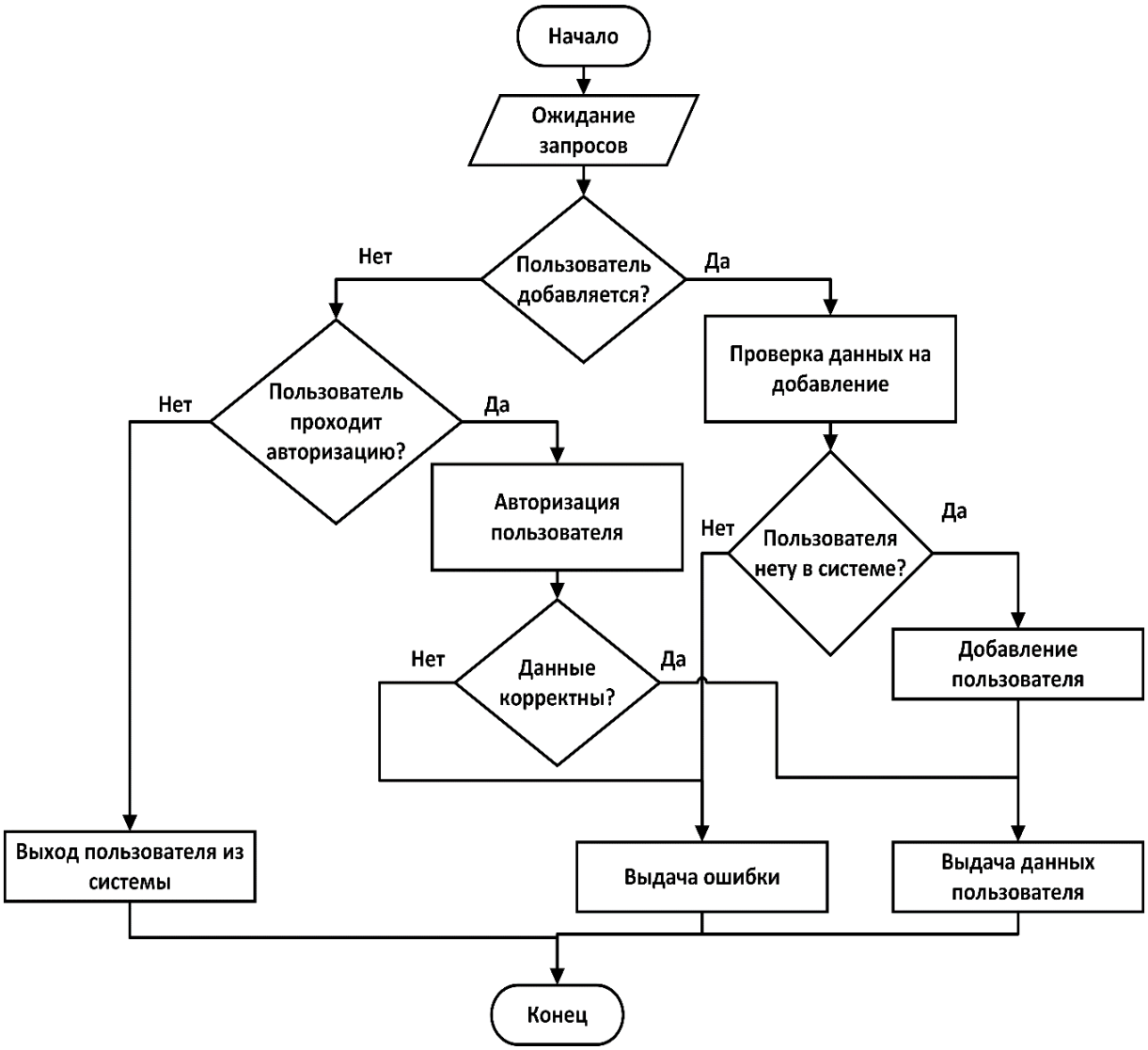

Рисунок 4 – Диаграмма серверного модуля

Этапы работы модуля приложения:

- этап 1 – ожидание ввода данных от пользователя для авторизации, по окончанию ввода перейти к этапу 2;

- этап 2 – передать данные серверному модулю и дождаться ответа. В случае положительного ответа – перейти к этапу 3, в ином – к этапу 1;

- этап 3 – загрузить редактор. После загрузки перейти к этапу 4;

- этап 4 - ожидать ввода от пользователя. В случае добавления изображения или любого инструмента на холст, перейти к этапу 5. Если элемент редактируется, перейти к этапу 6. В случае создание нового изображения, перейти к этапу 7. Если пользователь загружает собственной изображение, перейти к этапу 8. Если пользователь сохраняет работу, то перейти к этапу 9. Если пользователь выходит из системы, то перейти к этапу 10;

- этап 5 – добавить выбранный элемент на холст. Переход к этапу 4;

- этап 6 – применить вводные данные для выбранного элемента. Переход на этап 4;

- этап 7 – создать новый холст с размерами по умолчанию и белым фоном. Переход к этапу 4;

- этап 8 – добавить загруженное изображение на холст. Переход к этапу 4;

- этап 9 – подготовить данные холста к выдаче и предложить форму сохранения. После сохранения, переход к этапу 4;

- этап 10 – отправить запрос серверному модулю на выход. Переход к этапу 1.

Диаграмма деятельности данного модуля показана на рисунке 5.

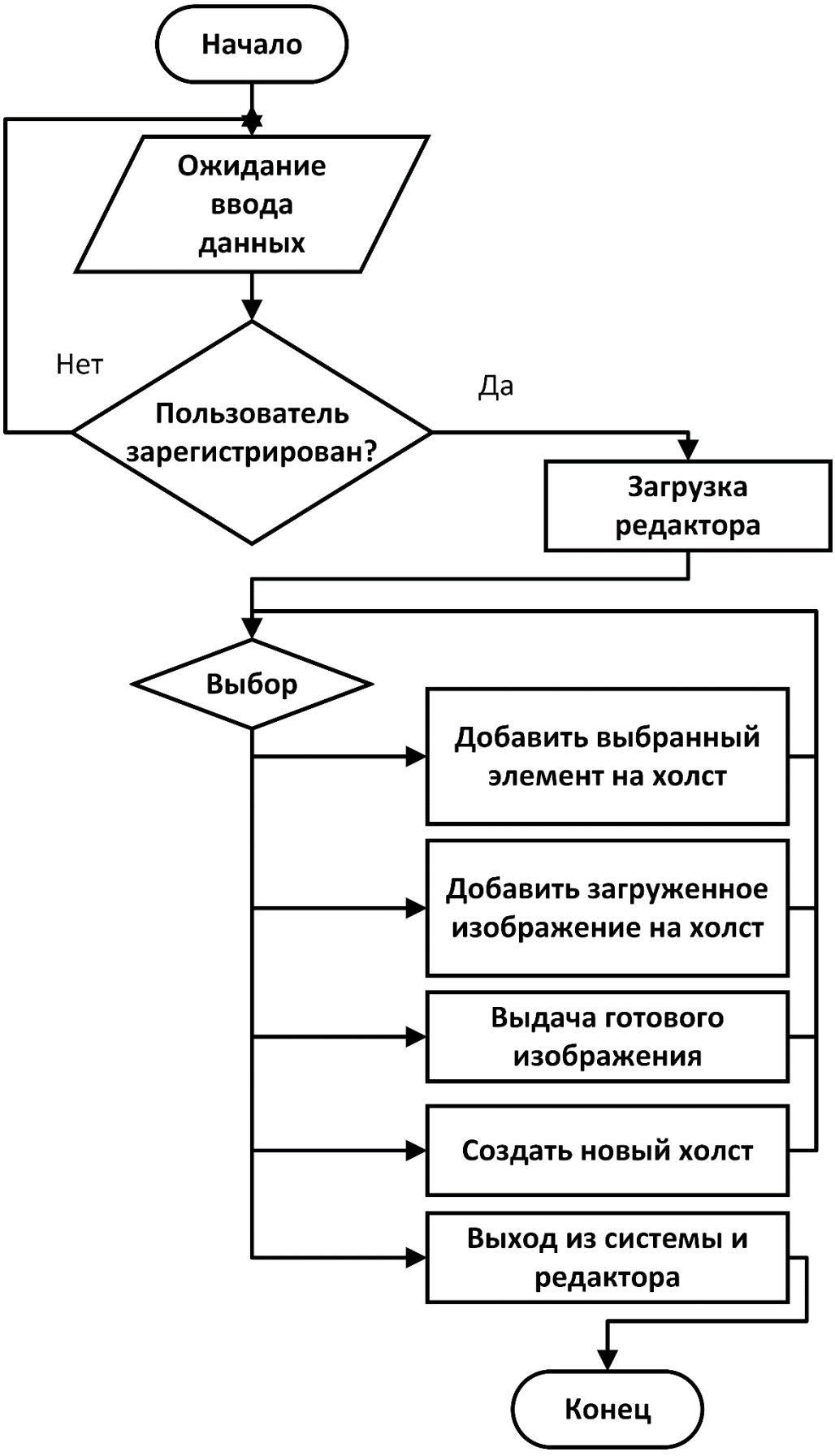

Рисунок 5 – Диаграмма приложения

2.3 Требования к программе

2.3.1 Общие требования

Из исходя вышеизложенного, проектируемое веб-приложение должно содержать следующие компоненты:

- модуль приложения;

- серверный модуль.

Данный программный продукт ввиду своей открытости может подвергаться модифицированию и улучшению на основе потребностей заинтересованных лиц, которые могут копировать кодовую базу и вносить свои изменения на усмотрение.

Пользователями информационной системы могут быть разные люди, в том числе и не имеющих профессиональных навыков в области дизайна.

Программный продукт разворачивается единожды на сервер со своим доменным именем, который после могут посещать большое число пользователей.

Требования к надежности оборудования и ПО:

- проведения комплекса мероприятия отладки, поиска и устранения ошибок;

- использование сертифицированных средств вычислительной техники, их комплектующих и средств передачи данных;

- обеспечение защиты от перехвата информации по техническому каналу;

- использование надёжных методов шифрования данных в базе данных;

- с целью повышения отказоустойчивости аппаратного оборудования необходима обязательная комплектация серверов источником бесперебойного питания с возможностью работы системы не мене 30 минут.

Требования к интерфейсу:

- интерфейс должен быть простым и понятным с низкой сложностью освоения и поощрять знания и умения людей, которые работали со схожим программным продуктом;

- удобство интерфейса позволяет пользователя быстрее работать;

- цветовая гамма должна быть приятной и простой, не нагружая глаз;

- возможность ввода данных по мере необходимости как с клавиатуры, так и с мыши.

2.3.2 Требования к лингвистическому обеспечению

Веб-приложение должно быть полностью на русском языке. Ввод данных пользователя производится только латинскими буквами и цифрами.

Для разработки приложения был выбран мультипарадигменный язык программирования – JavaScript.

2.3.3 Требования к информационному обеспечению

Код программного продукта минимизирован, поэтому при работе с приложением не требуется хранения большого объема данных или загрузки их через сети Интернет. В данном случае ресурсов компьютера считается достаточным.

2.3.4 Требования к программному обеспечению

Основой разрабатываемого веб-приложения является веб-браузер. Выбор браузера, на которой работает данная система, разнообразен, но лучше всего использовать Google Chrome компании Google последней версии, поскольку на ней была протестирована работа и имеет следующие достоинства:

- стабильная работа на операционных системах как для настольных компьютеров, так и мобильных;

- большая пользовательская база;

- имеет передовую поддержку стандартов HTML, CSS и JS;

- высокая производительность синтаксического анализатора JavaScript – V8.

Готовое изображение в формате PNG, который предлагает лучшее качество изображение при сравнительно небольших размерах. Название файла задается пользователем по желанию, по умолчанию – это «Новое изображение».

2.3.5 Требования к техническому обеспечению

Разрабатываемое веб-приложение не требует много места на компьютере или сильных вычислительных мощностей, ограничиваясь самыми минималь-

ными требованиями, поэтому программный продукт доступен для любого пользователя с выходом в Интернет и наличия веб-браузера. Таким образом, минимальными требования к ПК пользователей будут следующими:

– процессор Intel Pentium 4 серии или позднее с поддержкой наборов инструкций SSE2;

- ОС: Windows 7, 8, 8.1, 10 или позднее;

- объем оперативной памяти – 128 Мб;

- место на накопителе – 64 Мб;

- сетевое подключение с выходом в Интернет;

- наличие установленного веб-браузера – Google Chrome или другие.

3 ОПИСАНИЕ РАЗРАБОТАННОГО ВЕБ-ПРИЛОЖЕНИЯ

3.1 Обоснование выбора языка программирования, среды разработки и использованных библиотек

JavaScript – это прототипно-ориентированный мультипарадигменный язык программирования с динамической типизацией, который поддерживает объектно-ориентированный, императивный и декларативный стили программирования [9]. Язык постоянно развивается, поэтому синтаксис языка становится богаче с каждым годом. Изучение его или переход на него не составляет труда, так как язык очень хорошо документирован и имеет множество примеров.

JavaScript поддерживается многими браузерами по умолчанию как язык сценариев, поэтому каждый веб-браузер имеет свой подход к анализатору исходного кода.

Несмотря на поддержку и зачастую использование только в веб-браузерах на сайтах и веб-приложениях, данный язык программирования также можно использовать для разработки мобильных, серверных, прикладного программного обеспечения.

Процесс написания программного кода для веб-приложения проще чем на любом языке программирования ввиду наличия большого выбора подключаемых библиотек. Приложение структурируется по усмотрению разработчика, однако для данного веб-приложения исходные файлы сведены к компонентам, то есть каждый файл представляет собой отдельно логически связанную компоненту, которая потом может быть использована заново в любой части приложения.

Средой разработки был выбран Microsoft Visual Studio Code. Преимуществами данной среды являются:

-наличие в Visual Studio Code поддержки языка программирования JavaScript по умолчанию, не требуя от разработчика установки дополнительных инструментов для работы, а также замечательные инструменты для отладки приложений на данном языке.

- возможность добавления множества расширения из вкладки «Расширения», которые могут добавлять как поддержку нового языка программирования, дополнительно отладочных инструментов, так и настройки внешнего вида и интеграции распределенных систем управления версий;

- высокая скорость разработки за счёт удобных функций подсказок при написании, поддержки сокращений и популярный IntelliSense, который позволяет без запуска приложения анализировать код и проверять его на наличие ошибок, подчеркивая их и предлагая исправления;

- поддержка окна терминала в самой среде, а также наличие до нескольких экземпляров терминала для удобной работы разработчиком над приложением;

- наличие сохранения и загрузки конфигураций рабочего пространства для каждого проекта;

- высокая скорость работы и запуска по сравнению с другими популярными средами – Atom, Microsoft Visual Studio, Sublime Text.

Для написания модуля приложения использовалась библиотека ReactJS от компании Facebook. Эта JavaScript библиотека с открытым исходным кодом позволяет легко и удобно разрабатывать одностраничные веб-приложения. Помимо этого, цель React – предоставить высокую скорость, простоту и масштабируемость.

Удобство использования библиотеки React заключается в использовании, так называемого, JSX – совмещение HTML разметки и функциональности JavaScript в коде, что позволяет создавать многоразовые компоненты, поэтому ориентироваться в коде просто и логично. Другое же новшество, которое привносит библиотека, является понятие «состояние» — это текущее значение переменных в виртуальной объектной модели документа. Суть заключается в создании структуры документа, включая все дочерние элементы, в памяти компьютера (от того и виртуальная и не зависит от той, что есть в браузере). При изменении состояния одного из элемента, вычисляется лишь разница между новой структурой и старой, что позволяет оптимально обновлять объектную модель документа браузера [8].

Говоря о серверном модуле, то для его разработки была выбрана программная платформа Node.js, которая работает с JavaScript приложениями.

Плюсами платформы является:

- использование движка V8 от Google, который используется в веб-браузере Google Chrome;

- легко осваиваемый благодаря богатой документации и минимального труда для разработки простого прослушиваемого сервера;

- хорошо масштабируется ввиду поддержки специализированных методов для работы с масштабированием;

- имеет хорошую поддержку документно-ориентированной системы управления базами данных mongoDB;

- умеет взаимодействовать с устройствами ввода-вывода;

- поддержка внешних библиотек для расширения функционала;

- основан на событийно-ориентированном и асинхронном программировании, которые не блокируют ввод или вывод [7].

3.2 Описание модулей веб-приложения

При разработке веб-приложения были выделены два модуля, на основе которых созданы были следующие компоненты:

Для модуля приложения:

- App объединяет компоненты Login и ImageEditor;

- Login компонент для входа и регистрации пользователей;

- ImageEditor обеспечивает пользовательский интерфейс: связывая компоненты ImageHeader, ImageConfig, ImageFooter, ImageItems, ImageTitle и Canvas;

- ImageTitle представляет верхнюю строку заголовка веб-приложения, приветствия пользователя и кнопки выхода из системы;

- ImageConfig предоставляет настройки холста, загрузки изображения, просмотра элементов на холсте и индивидуальную настройку (в случае если выбран) инструмента;

- ImageFooter отвечает за дополнительные инструменты выбора, захвата и масштаба внизу рабочей области;

- ImageItems реализует список доступных инструментов для редактирования на холсте;

- ImageHeader обеспечивает верхнюю строку опций для работы с изображениями и элементами на холсте;

- Canvas реализует рабочую область для работы с изображением и отрисовку всех изменений.

Для серверного модуля:

- Index отвечает за совместную работу компонентов Routes и Models и прослушивание порта 9000 на запросы от модуля приложения;

- Routes отвечает за все возможные (имеются ввиду те, что запрограммированы) пути обращения к модулю;

- Models предоставляет интерфейс для работы с базой данных. Интерфейс представляется в виде «моделей», которые задаются разработчиком. В серверном модуле их две – History и User. Речь о них пойдет в следующем разделе.

3.3 Описание БД

Для работы серверного модуля и хранения данных пользователей была разработана база данных mongoDB, которая предоставляет прямую работу с базой на языке JavaScript и легко согласовывать данные между серверным модулем и БД ввиду моделирования данных в JSON-нотации. В базе данных хранятся две сущности:

1. Сущность «User» хранит данные о пользователях, имеющих учетные записи для входа в программу. Здесь сохраняются уникальный идентификатор, присваиваемый самой базой данных, имя пользователя, электронная почта, пароль для входа, дата создания учетной записи.

Таблица 1 – Спецификация атрибутов сущности «User»

Название атрибута	Описание атрибута	Тип данных	Пример атрибута
_id	Ключевой уникальный идентификатор, позволяющий однозначно определить пользователя	ObjectId	21asc2a23d
username	Имя пользователя	Текстовый	Username1
Email	Электронная почта	Текстовый	sample@mail.ru
password	Пароль пользователя	Текстовый	Password1
createdAt	Дата создания учетной записи	Дата Время	04/01/2019 12:30

2. Сущность «History» хранит данные, которые представляют собой историю совершенных действий пользователем с интерфейсом и серверным модулем. В этой таблице содержатся уникальный идентификатор истории, идентификатор пользователя, дата совершения действия или входа в систему, ip-адрес, с которого произошел вход и уникальный идентификатор пользователя, от имени которого совершилось действие.

Используемая база данных используется в качестве хранилища данных. Связи между сущностями в ней имеется, где история связана с пользователем. В качестве первичного ключа выбраны поля, которые не повторяются и являются уникальными. В таблице «User» и «History» первичный ключ генерируется автоматически базой данных. На рисунке 6 показано физическая модель базы данных.

Таблица 2 – Спецификация атрибутов сущности «History»

Название атрибута	Описание атрибута	Тип данных	Пример атрибута
_id	Ключевой уникальный идентификатор, позволяющий однозначно определить экземпляр истории	ObjectId	21asc2a23d

Продолжение таблицы 2

userId	Внешний ключ, позволяющий однозначно связать пользователя с экземпляром истории	ObjectId	21asc2a23d
Action	Совершенное действие в системе или вход	Текстовый	login
loginDate	Время действия или входа в систему	Дата Время	04/01/2019 12:30
Ip-адрес	Сетевой адрес клиента	Текстовый	169.254.18.235

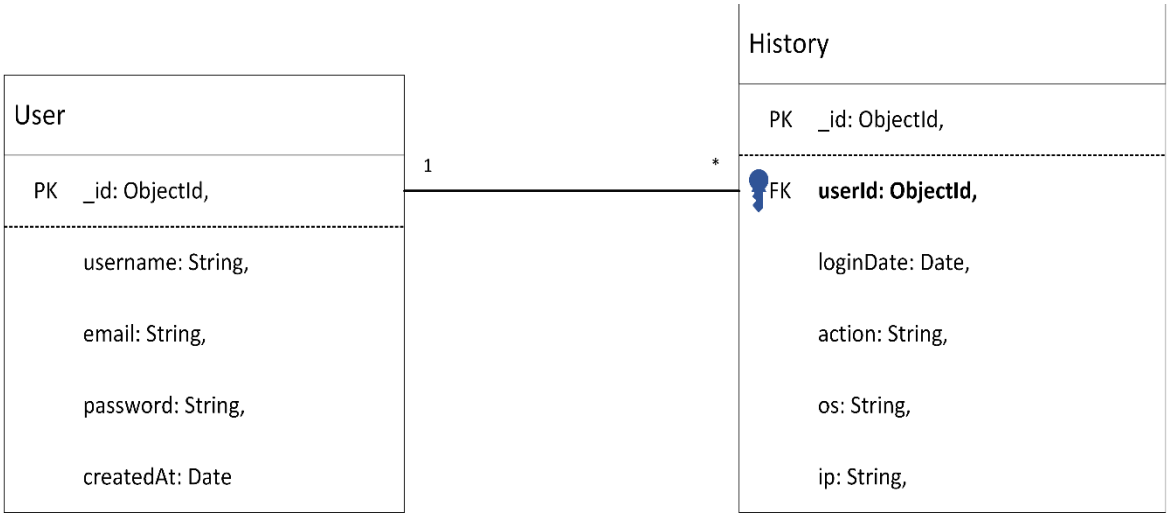

Рисунок 6 – Физическая модель базы данных

3.4 Тексты запросов

При использовании модуля приложения пользователя встречает форма регистрации, в которую нужно ввести предпочитаемые данные учетной записи (имя пользователя, логин и пароль) для создания пользователя и автоматического входа в систему. Текст запроса, который используется в серверном модуле для добавления пользователей, представлен на рисунке 7.

```
router.post('/create', async (req, res) => { //creates users
  console.log('Получен запрос на регистрацию...');
  let user = await req.context.models.User.findOne({
    email: req.body.email,
  });
  console.log(user);
  if (!user) {
    console.log(req.body);
    await req.context.models.User.create({
      username: req.body.username,
      email: req.body.email,
      password: req.body.password,
      createdAt: req.body.date,
    }, (err, result) => {
      if (err) { res.json('Ошибка регистрации'); }
      console.log(req.body.data);
      const data = new req.context.models.History({
        loginDate: req.body.date,
        action: req.body.data.action,
        os: req.body.data.os,
        device: req.body.data.ua,
        ip: req.ip,
        userId: result._id,
      });
      data.save();
      console.log('Регистрация успешно выполнена!');
      console.log(result);
      res.status(200).redirect(`/users/${result._id}`);
    });
  } else {
    console.log('Пользователь имеется в БД.');
    res.sendStatus(403);
  }
});

```

Рисунок 7 – «Обработка запроса на создание пользователя»

Регистрация пользователя в системе происходит путем его добавления, если его изначально не было. Данные о пользователе после регистрации выдаются модулю приложения. Помимо этого, пользователь может быть зарегистрирован в системе, поэтому при попытке авторизации, серверный модуль должен обработать его соответствующе. Тексты таких запросов представлены на рисунках 8 и 9.

```
74  router.get('/:userId', async (req, res) => {
75    console.log('Получен переход от регистрации...');
76    await req.context.models.User.findOne({
77      _id: req.params.userId,
78    }, (err, result) => {
79      if (err) { res.json('Ошибка получения данных пользователя!') }
80      console.log('Выдача данных пользователя.');
81      console.log(result);
82      res.status(200).json(result);
83    });
84  });
85
```

Рисунок 8 – «Выдача данных о пользователе после регистрации»

```
16  router.post('/login', async (req, res) => { //login users
17    let user;
18    console.log('Получен запрос на вход пользователя...');
19    await req.context.models.User.findOne({
20      email: req.body.email,
21      password: req.body.password,
22    }).then(fetchUser => {
23      console.log("Вход пользователя "+ fetchUser.email + " в " + new Date());
24      const data = new req.context.models.History({
25          loginDate: req.body.data.date,
26          action: req.body.data.action,
27          os: req.body.data.os,
28          device: req.body.data.ua,
29          ip: req.ip,
30          userId: fetchUser._id,
31      })
32      data.save();
33      console.log(data);
34      console.log(fetchUser);
35      res.json(fetchUser);
36    }).catch(() => res.status(401).json({err: 'Введены неверно данные.'}));
37  });
38
```

Рисунок 9 – «Выдача данных о пользователе после их входа»

Все обращаемые запросы на запрос к данным пользователя отслеживаются и заносятся в сущность модель конкретного пользователя (при наличии в системе).

3.5 Руководство пользователя

Программа начинает свою работу после обращения к ней по унифицированному указателю ресурсу или, при развертке локально на компьютере, по localhost:3000. Для запуска приложения в последнем случае необходимо установка и запуск скриптов в папках «server» и «client».

Для установки необходимых зависимостей, надо заранее установить NodeJS 10.16.0 LTS и mongoDB Community Server версии 4.0.9. Важно, чтобы сперва запустился серверный модуль, а только после модуль приложения. Поэтому рекомендуемые шаги для установки и запуска веб-приложения следующие:

1. Запустить терминал и перейти в папку веб-приложения.

2. Перейти в папку «server»

3. Выполнить команду npm install. Дождаться выполнения команды.

4. Запустить сервер командой npm start.

5. Так как в текущем окне терминала работает серверный модуль, стоит открыть новое окно терминала и перейти в папку веб-приложения.

6. Перейти в папку «client»

7. Установить зависимости, выполнив команду npm install.

8. Дождавшись установки, запустить модуль приложения командой npm start.

9. Дождавшись запуска, по умолчанию откроется браузер с ссылкой на запущенный сайт.

После загрузки веб-приложения, пользователя встречает форма авторизации и регистрации. Обе формы требуют от пользователя ввода электронной почты, пароля, а для регистрации еще и имя пользователя. Форма входа представлена на рисунке 10, а на рисунке 11 – форма регистрации.

Рисунок 10 – «Форма входа в аккаунт»

Рисунок 11 – «Форма регистрации»

Введя данные учетной записи, пользователь попадает в основную часть приложения – редактор изображения.

Сверху слева размещается название программы, по другую сторону отображается приветствие пользователя и кнопку выхода из системы. Ниже названия размещается быстрая панель инструментов, которая предлагает создать новое изображение, сохранить изображение, панель управления размещения выбранного элемента относительно других, выравнивание текста, группировка элементов, разгруппировка, обрезание загруженных изображений, клонирование элемента, удаления, кнопки назад и вперед, а также переключатель на темную тему. На панели слева от экрана отображены инструменты для редактирования изображения. Правая панель отвечает за настройку выбранного инструмента, загрузки изображения, свойств холста и списка элементов на холсте. По центру расположена рабочая область приложения, на которой отображается весь результат работы. Снизу расположены инструменты выбора, захвата и масштабирования рабочей области. Интерфейс приложения отображен на рисунке 12.

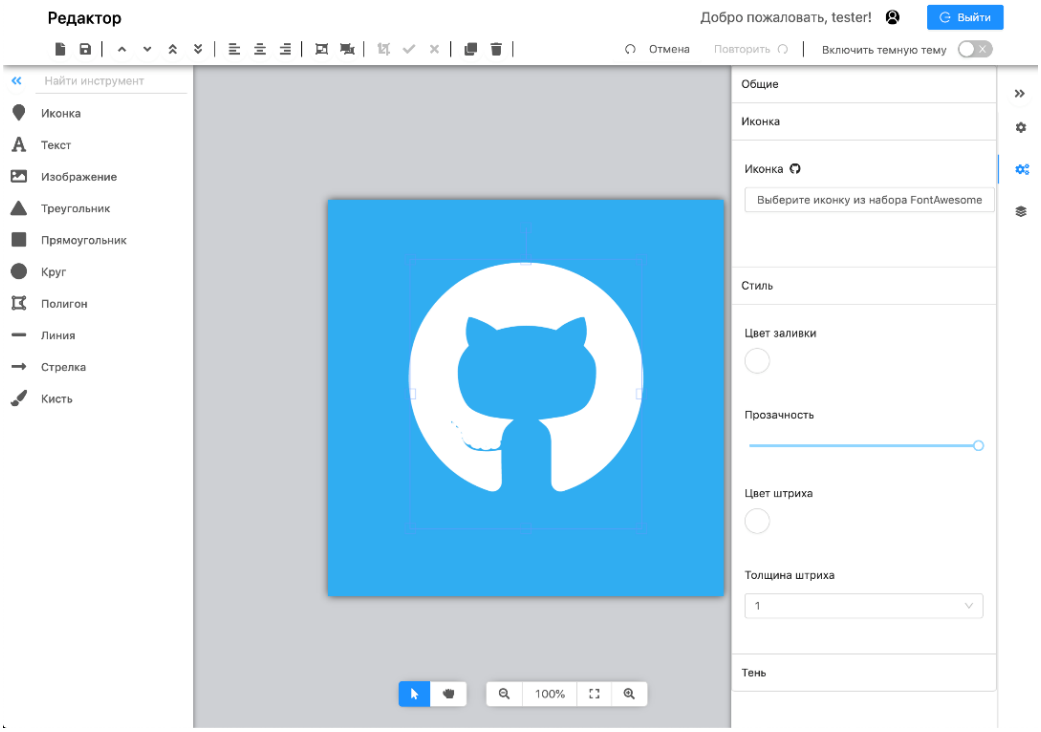

Рисунок 12 – «Редактор»

При переключении темы, весь фон интерфейса меняет свой цвет на темный, а элементы для контраста становятся белыми, это позволяет значительно

уменьшить наличие белого цвета на экране и нагрузку на глаза. Рисунок 13 демонстрирует темную тему.

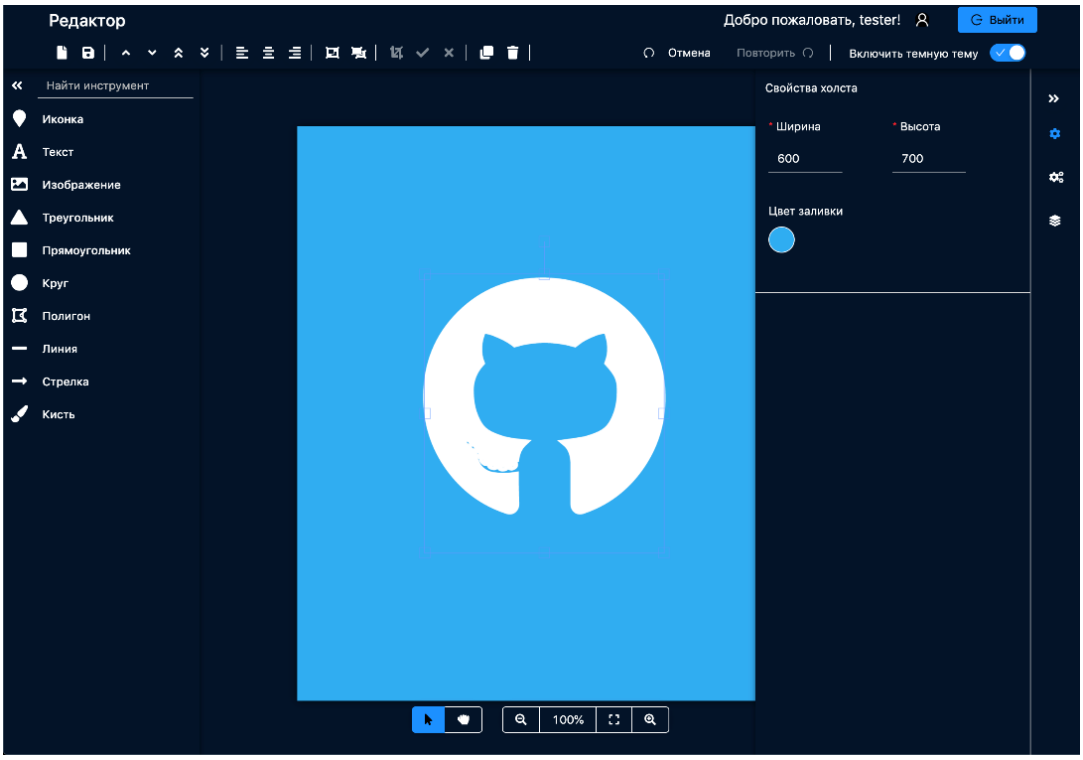

Рисунок 13 – «Редактор в тёмной теме»

4 УГРОЗЫ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ ПРИЛОЖЕНИЯ

4.1 Модели угроз

Информация является важным ресурсом любой информационный системы, поэтому обеспечение сохранности и доступности данных представляет собой одну из важных задач при разработке и функционировании ИС.

Безопасность информационной системы характеризируется целостностью и секретностью информации, чтобы обеспечить её нормальное функционирование и сохранность конфиденциальных данных. Для реализации конфиденциальности, достоверности, доступности и целостности информации необходимо внедрить механизмы предотвращения несанкционированного доступа с целью кражи, модификации и уничтожения информации.

Стоит отметить, что под **угрозой** понимается потенциально возможное случайное или преднамеренное событие, процесс или действие, приводящее к нарушению конфиденциальность, целостности, доступности информации и поддерживающей её инфраструктуры, нанося ущерб владельцу или пользователю информации.

При решении вопроса защищённости информационной системы прибегают к **моделям угроз**, **модель возможного нарушителя** и **мерам по предотвращению действий злоумышленника**:

- Модель угрозы – это свойства или характеристика угроз информационной безопасности, которая может быть описана физически, математически или визуально;

- Модель возможного нарушителя – это человек, который случайно или намеренно нарушил информационную безопасность в следствии своих действий;

- Меры предотвращения действия злоумышленника – это комплекс организационных, правовых, технических мер по обеспечению информационной безопасности.

Говоря о разработанном веб-приложения, то можно выделить следующие модели угроз:

- Атака «отказ в обслуживании», цель которой вывести ИС из рабочего состояния путём длительной нагрузки на вычислительные мощности системы из-за чего пользователи не могут получить доступ к системе;

- Сетевая атака – использование программных и/или технических средств для реализации угроз несанкционированного доступа к информации через сетевые протоколы;

- Инъекции – внедрение в запросы к базе данных кода, позволяющий скомпрометировать информацию в базе данных – от получения до её уничтожения;

- Незащищённость важных данных – получение информации через уязвимости в ПО информации о пользователях;

- XSS (Cross-Site Scripting) – подстановка скрипта из ненадёжных источников, заменяя те, что используются в веб-приложении;

- Использование компонентов или пакетов с известными уязвимостями, которые есть в публичном доступе в интернете и хорошо документированы;

- Перехват информации;

- Подбор паролей.

4.2 Модель нарушителя

Говоря о модели нарушителя, вводится понятие **нарушитель** – это лицо, умышленно или неумышленно предпринявшее попытку реализации угрозы информационной безопасности. Для атаки принимаются любые действия нарушителем, которые приводят к реализацию угрозы путём использования уязвимостей программного или аппаратного обеспечения, применяющиеся для функционирования ИС.

Под **уязвимостью** понимается любая характеристика или свойство сетевого, или системного компонента ИС, использование которого нарушителем может привести к реализации угрозы.

Для веб-приложения подразделяются следующие категории нарушителей:

- пользователи ИС – зарегистрированные пользователи ИС, осуществляющие доступ к ресурсам ИС;

- временные пользователи ИС – незарегистрированные пользователи;

- программисты – программисты-разработчики ПО, которые осуществляют поддержку и администрирование работоспособности ИС;

- злоумышленник – лицо (группа лиц), не являющееся зарегистрированным пользователем, не имеющим полномочий для доступа к ресурсам ИС, но пытающийся получить доступ к информации в системе.

Источниками угроз ИБ можно разделить на внутренние и внешние. К **внутренним источникам** угроз относят:

- аппаратное обеспечение, применяющиеся в ИС;

- программное обеспечение;

- сетевое оборудование;

- системы жизнеобеспечения.

К **внешним источникам** угроз информационной безопасности информационной системы относят:

- нарушители информационной безопасности ИС;

- форс-мажорные обстоятельства.

По мотивации атаки на информационную систему можно разделить на две категории:

- преднамеренные угрозы связаны с корыстными целями нарушителей.

- случайные угрозы вызваны ошибками в работе программного обеспечения информационной системы.

Из исходя вышесказанного, для разработанного веб-приложения выделяются следующие модели угроз информационной безопасности:

- угрозы ввиду возникновения ошибок в системном и функциональном программном обеспечении компонентов системы;

- угрозы из-за предумышленных или непредумышленных действий человека;

- угрозы, которые возникают вследствие физических повреждений, отказов и неисправностей технических средств системы, её отдельных компонентов и вспомогательных коммуникаций;

- угрозы, возникающие вследствие форс-мажорных обстоятельств.

В таблицах 3, 4, 5 и 6 приведены модели угрозы.

Таблица 3 – Модели угроз, возникающие в системном и функциональном программном обеспечении компонентов системы

Вид угроз	Описание угрозы	Источник
Отказ системного ПО	Отказ системного ПО из-за наличия ошибок, влекущих возникновение уязвимостей и сбоев в работе ПО	Программные средства
Отказ прикладного ПО	Отказ прикладного ПО вследствие наличия ошибок, влекущих возникновение уязвимостей и сбоев в работе системы	Программные средства

Таблица 4 – Модели угроз из-за предумышленных или случайных действий человека

Вид угроз	Описание угрозы	Категории нарушителя
Неправомерные действия авторизованных пользователей	Использование нарушителем учетной записи, к которой ему разрешен доступ, в неразрешенных целях, в том числе неправомерные действия	Программисты, пользователи.
Отказ в обслуживании	Выполнение намеренных действий, направленных на возникновение отказа в обслуживании в системах, приложениях, базах и сетях передачи данных	Злоумышленники, программисты, пользователи
Внедрение вредоносного программного обеспечения	Внедрение вредоносного ПО: «троянских коней», «червей», «логические бомбы» - приводящее к сбою или нарушению в работе компонентов ИС, а также получению полного контроля над уязвимой системой	Злоумышленники, пользователи, временные пользователи, программисты

Продолжение таблицы 4

Подмена имени авторизованными пользователями	Получение доступа (например, с помощью использования чужой учетной записи) авторизованными пользователями к информации, доступ к которой им запрещен	Пользователи, временные пользователи
Подмена имени пользователя посторонними лицами	Получение посторонними лицами доступа (под именем авторизованного пользователя) к информации	Злоумышленники, временные пользователи, пользователи
Ошибка пользователя	Совершение ошибок пользователями при работе с приложениями	Пользователи, временные пользователи

Таблица 5 – Модели угроз у технических средств

Вид угроз	Описание угрозы	Источник
Отказы и сбои серверного оборудования	Отказ в работе серверов в результате влияния различных факторов	Технические средства
Отказы и сбои дисковых массивов	Отказ в работе дисковых массивов в результате влияния различных факторов	Технические средства
Отказы и сбои сетевого оборудования	Отказ в работе сетевого оборудования в результате влияния различных факторов	Технические средства

Таблица 6 – Модели угроз в форс-мажорных обстоятельствах

Вид угроз	Описание угрозы	Источник
Отказ системы энергоснабжения	Отказ подачи электропитания в помещение	Технические средства

Продолжение таблицы 6

Отказ системы кондиционирования	Отказ системы воздушного кондиционирования, приводящий к приостановке работы вследствие выхода рабочих температур за предельно допустимые рамки	Технические средства
Пожар	Повреждение огнём физических средств, составляющих систему, включая документацию и магнитные носители данных	Форс-мажорные обстоятельства
Затопление	Повреждение водой физических средств, составляющих систему, включая документацию и магнитные носители данных	Форс-мажорные обстоятельства
Стихийные бедствия	Землетрясения, сели, ураганы, наводнения и прочие катастрофические природные явления	Форс-мажорные обстоятельства

4.3 Меры по предотвращению действий злоумышленника

Для защиты от вышеперечисленных моделей угроз, предлагается придерживаться следующих мер:

Для моделей угроз, возникающих при работе программного обеспечения:

- использование последних версий компонентов и пакетов, полученных с надёжных источников;

- быстрое устранение ошибок программистами;

- наличие нескольких запущенных экземпляров приложения на различных серверах, каждый из которых может использоваться в качестве резервного.

Для моделей угроз, возникающих по вине человека:

- ограничение прав пользователей;

- использовать серверов-разгрузчиков, которые будут перенаправлять пользователей на сервер с наименьшей нагрузкой;

- использование контейнеров, которые изолированы от ОС. Это разрешает многие вопросы по конфигурации сервера, например, получение удалённого root доступа;

- ограничение и изолирование на сервере функций ОС, портов, протоколов. Не использовать стандартную учётную запись;

- шифрование данных клиентов в базе данных, в которой каждая учетная запись шифруется при помощи заданного в учетной записи пароля;

- использование надёжных паролей;

- безопасные протоколы HTTPS, TLS для передачи персональных данных;

- для DDoS использовать возможности, предоставляемые интернет платформ, на базе которых размещается информационная система.

Для моделей угроз технических средств и в результате форс-мажора:

- наличие персонала, который будет отслеживать работоспобность серверов;

- создание резервных копий;

- установка веб-приложений на нескольких серверах.

5 БЕЗОПАСНОСТЬ НА ПРЕДПРИЯТИИ

5.1 Безопасность

Безопасность рабочего места является важным аспектом трудовой деятельности работников, для которых это место предназначено. Поэтому при взаимодействии с персональной электронно-вычислительной машиной или видеодисплейных терминалов (далее ПЭВМ и ВДТ) стоит обратить внимание на снижение рисков и травм оператору, ведь при работе с ПЭВМ и ВДТ существуют ряд опасных и вредных производственных факторов, значимые из которых следующие:

- электротравма ввиду отсутствия заземления оборудования. При работе с ПК используется переменный ток частотой 50 Гц напряжением 220 В для питания схем и узлов дисплея;

- высокий уровень напряженности полей электрического и магнитного в широком диапазоне частот;

- повышенный уровень шума вентиляторов охлаждения ПК и принтера;

- не соответствие санитарным нормам визуальные параметры дисплеев, особенно имеющих величину зерна (пиксель) 0,3 мм и более, частоту кадровой развертки (50-75) Гц;

- избыточные энергетические потоки сине-фиолетового света от экрана дисплея в видимом диапазоне длин ЭМ-волн, которые снижают четкость восприятия изображения глазом;

- повышенный уровень запыленности воздуха рабочей зоны от внешних источников;

- физическое перенапряжение из-за нерациональной организации рабочего места, что существенно даёт напряжение мышцам позвоночника, ног, рук, шеи и глаз;

- повышенные зрительные нагрузки и адинамия глазных мышц, то есть их малая подвижность при высоком статическом в течение длительного времени, что вероятно становится причиной различных глазных заболеваний (например,

снижение остроты зрения, близорукость, спазм аккомодации – потеря возможности глазных мышц сокращаться);

- внешние постоянно действующие экологические факторы: наличие в воздухе вредных веществ (озона, аммиака, углекислый газ, серы, солей тяжелых металлов и органических соединений).

Таким образом, если перечисленные факторы воздействуют на человека, то здоровье человека находится под угрозой. Если организм человека не совсем здоров, то такое отрицательное воздействие усугубляется [4, с. 6].

Говоря о работе с программным обеспечением за ПК, человек сталкивается с работой графического интерфейса пользователя, который визуально предоставляют информацию и кнопки управления или взаимодействия с ними [3, с. 125].

При разработке интерфейсов у дизайнеров существует понятие «дизайн взаимодействия с пользователем», который повышает удовлетворенность работы пользователем программного обеспечения за счет улучшения доступности, эргономики и наслаждения при взаимодействии между пользователем и продуктом.

Таким образом, стоит обратить внимание на **эргономику** интерфейса. Эргономикой называют изучение человека и его деятельности в условиях современного производства с целью совершенствования орудий труда, улучшения условий работы и оптимизации трудового процесса [5].

5.1.1 Эргономичность интерфейса

Эргономичностью называют эффективность инструмента производства или системы в эргономике.

К эргономичности интерфейса относят удобство использованием пользователем программного обеспечения. Для оценки удобства существуют критерии эргономичности интерфейса: интуитивность (естественность), непротиворечивость (последовательность), визуализация, система навигации, гибкость, поддержка пользователя.

Интуитивность (естественность) – это свойство программного продукта адаптироваться под требования пользователя, а именно:

- общение происходит при помощи языка пользователя (в разработанном веб-приложении используется русский язык, а также поддержка английского языка);

- отсутствуют жёсткие требования к порядку ведения диалога пользователя с машиной;

- не требуется предварительная обработка данных перед вводом их пользователем в систему (это значительно влияет на взаимодействие и быстродействие, исключая появления ошибок).

Непротиворечивость или последовательность ведения диалога гарантирует единство общих принципов работы с системой. Данный критерий содержит:

- последовательность в интерпретации команд;

- последовательность в использовании форматов данных – в одном формате должны представляться аналогичные;

- последовательность в размещении информации на экране –информативность сообщения, должна предоставляться пользователю по степени важности (предупреждение об ошибке появится в центре экрана, а вспомогательная информация в углу).

Выделение элементов интерфейса актуализирует внимание пользователя на конкретной информации:

- цвет (создание интерфейсов, более интересных для пользователя. Он используется для группировки информации, выделения различий между информацией, выделения простых сообщений (ошибки, состояния));

- форма (вид символа, шрифт, начертание, размер);

- окружение (Поддержка пользователя во время диалога — это мера помощи, которую диалог оказывает пользователю при его работе с системой. Она включает в себя:

- инструкции пользователю – необходимы для направления пользователя в нужную сторону, подсказок и предупреждений для выполнения необходимых

действий на пути решения задачи. Инструкции обеспечены в форме диалога, экранных заставок, справочной информации и т.п. Они могут предложить пользователю: выбрать из предложенных альтернатив некую опцию или набор опций; ввести некоторую информацию; выбрать опцию из набора опций, которые могут изменяться в зависимости от текущего контекста; подтвердить фрагмент введенной информации перед продолжением ввода;

- подтверждение действий системы - используется, чтобы пользователь мог убедиться, что система выполняет, выполнила или будет выполнять требуемое действие;

- сообщения об ошибках - должны объяснить, в чем ошибка, и указать, как ее исправить.

Гибкость диалога — это мера того, насколько хорошо диалог соответствует различным уровням подготовки и производительности труда пользователя. При этом диалог может подстраивать свою структуру или входные данные. Гибкость диалога проявляется в способности диалоговых систем адаптироваться либо с помощью пользователя, либо самостоятельно к любому возможному уровню подготовки оператора. подчёркивание, рамки, инвертированное изображение).

Для эргономичности веб-приложения был разработан интерфейс с минималистичным дизайном, который не будет нагружать пользователя своей сложностью, позволяя сфокусироваться на работе, и поддержкой темной темы, создание которой уменьшает нагрузку на глаза в темных помещениях. Пользователь при наведении на любой из символов может видеть текстовые подсказки, которые помогают ориентироваться в интерфейсе и легче осваивать работу. На рисунке 14 показан редактор в тёмной теме.

Однако приложение можно доработать для доступности и поддержки людей с ограниченным возможностями, а именно: слабовидящих и с цветовой слепотой.

Для слабовидящих рекомендуется: установить крупный размер шрифта, расстояние между буквами большое, использование контрастных цветов для

легко различия визуальных элементов, увеличение размеров символов минимум в 2 раза.

Для людей с цветовой слепотой стоит использовать палитру с чётко различаемыми цветами, использование настраиваемых цветов для текста, фона, визуальных элементов под каждый тип цветовой слепоты. Также рекомендуется ношения специальных очковых линз, корректирующих цветовосприятие.

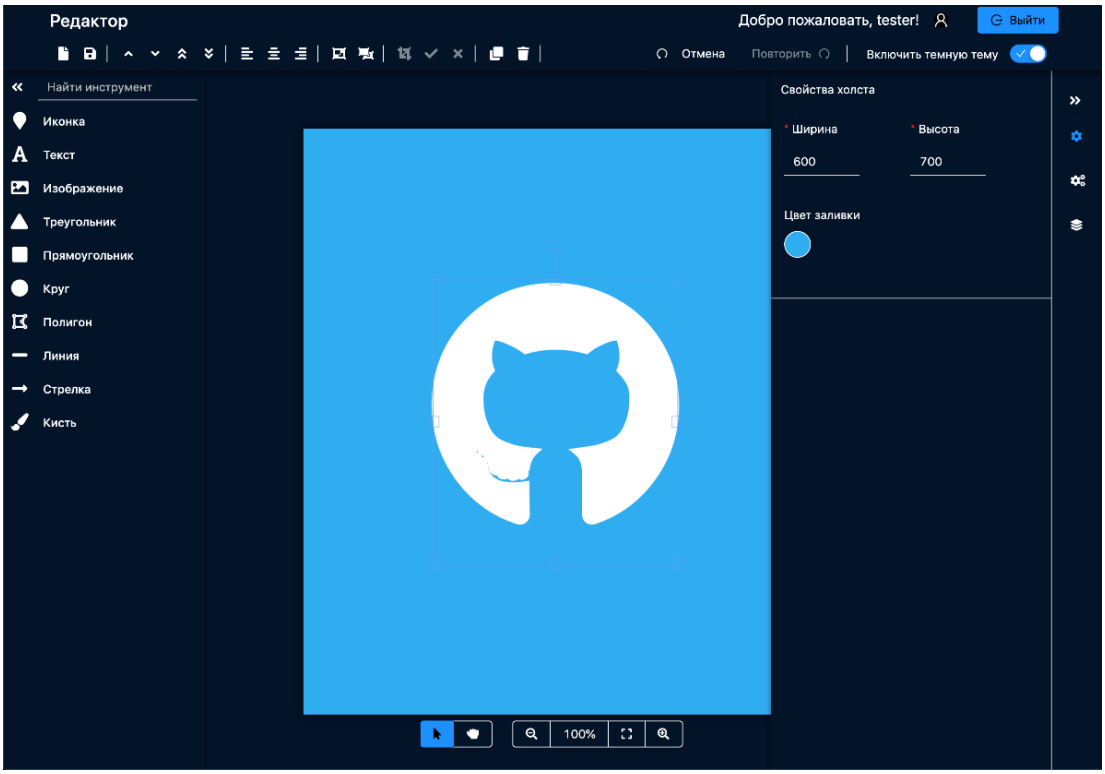

Рисунок 14 – Интерфейс пользователя.

5.1.2 Эргономичность рабочего места

Помимо интерфейса, важна и эргономичность рабочего места человека, поэтому предъявляются следующие требования к ПЭВМ для обеспечения эргономичности:

- ПЭВМ должны соответствовать требованиям настоящих санитарных правил, и каждый их тип подлежит санитарно-эпидемиологической экспертизе с оценкой в испытательных лабораториях, аккредитованных в установленном порядке;

- концентрации вредных веществ, выделяемых ПЭВМ в воздух помещений, не должны превышать предельно допустимых концентраций (ПДК), установленных для атмосферного воздуха;

- конструкция ПЭВМ должна обеспечивать возможность поворота корпуса в горизонтальной и вертикальной плоскости с фиксацией в заданном положении для обеспечения фронтального наблюдения экрана ВДТ. Дизайн ПЭВМ должен предусматривать окраску корпуса в спокойные мягкие тона с диффузным рассеиванием света. Корпус ПЭВМ, клавиатура и другие блоки, и устройства ПЭВМ должны иметь матовую поверхность с коэффициентом отражения 0,4-0,6 и не иметь блестящих деталей, способных создавать блики;

- конструкция ВДТ должна предусматривать регулирование яркости и контрастности.

Требования к освещению на рабочих местах, оборудованных ПЭВМ:

- естественное и искусственное освещение должно соответствовать требованиям действующей нормативной документации. Окна в помещениях, где эксплуатируется вычислительная техника, преимущественно должны быть ориентированы на север и северо-восток. Оконные проемы должны быть оборудованы регулируемыми устройствами типа: жалюзи, занавесей, внешних козырьков и др.;

- рабочие столы следует размещать таким образом, чтобы видеодисплейные терминалы были ориентированы боковой стороной к световым проемам, чтобы естественный свет падал преимущественно слева;

- освещенность на поверхности стола в зоне размещения рабочего документа должна быть 300-500 лк. Освещение не должно создавать бликов на поверхности экрана. Освещенность поверхности экрана не должна быть более 300 лк;

- общее освещение при использовании люминесцентных светильников следует выполнять в виде сплошных или прерывистых линий светильников, расположенных сбоку от рабочих мест, параллельно линии зрения пользователя при рядном расположении видеодисплейных терминалов. При периметральном расположении компьютеров линии светильников должны располагаться локализованно над рабочим столом ближе к его переднему краю, обращенному к оператору.

Общие требования к организации рабочих мест пользователей ПЭВМ:

- при размещении рабочих мест с ПЭВМ расстояние между рабочими столами с видеомониторами (в направлении тыла поверхности одного видеомонитора и экрана другого видеомонитора), должно быть не менее 2,0 м, а расстояние между боковыми поверхностями видеомониторов - не менее 1,2 м;

- рабочие места с ПЭВМ при выполнении творческой работы, требующей значительного умственного напряжения или высокой концентрации внимания, рекомендуется изолировать друг от друга перегородками высотой 1,5-2,0 м;

- экран видеомонитора должен находиться от глаз пользователя на расстоянии (600-700) мм, но не ближе 500 мм с учетом размеров алфавитно-цифровых знаков и символов;

- конструкция рабочего стула (кресла) должна обеспечивать поддержание рациональной рабочей позы при работе на ПЭВМ, позволять изменять позу с целью снижения статического напряжения мышц шейно-плечевой области и спины для предупреждения развития утомления. Тип рабочего стула (кресла) следует выбирать с учетом роста пользователя, характера и продолжительности работы с ПЭВМ. Рабочий стул (кресло) должен быть подъемно-поворотным, регулируемым по высоте и углам наклона сиденья и спинки, а также расстоянию спинки от переднего края сиденья, при этом регулировка каждого параметра должна быть независимой, легко осуществляемой и иметь надежную фиксацию;

- модульными размерами рабочей поверхности стола для ПЭВМ, на основании которых должны рассчитываться конструктивные размеры, следует считать: ширину 800, 1000, 1200 и 1400 мм, глубину 800 и 1000 мм при нерегулируемой его высоте, равной 725 мм.

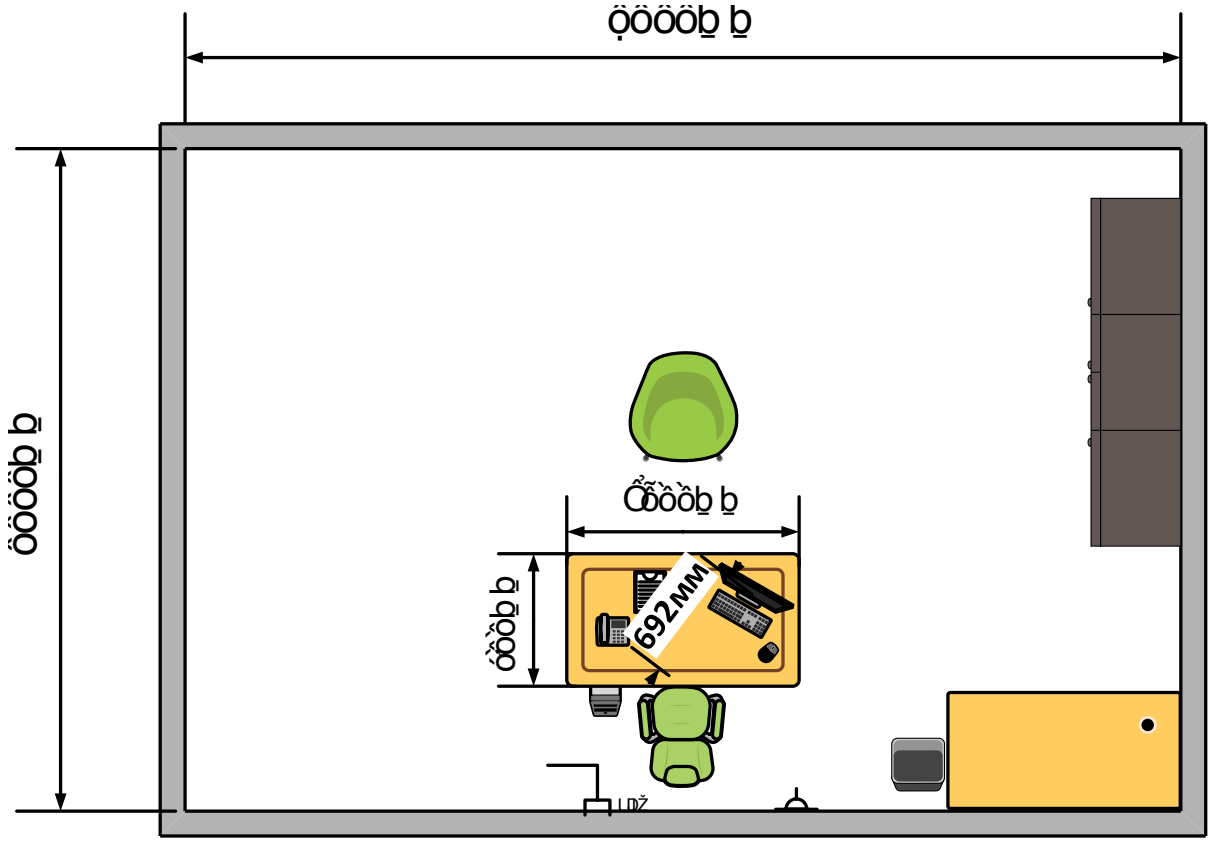

Рисунок 15 – Эргономичное расположение рабочего места

Как видно на рисунке 15 и рисунке 17, рабочая поверхность, расположение ПЭВМ относительно окна и относительно человека соответствует норме.

Расположение ламп указано на рисунке 16. Они находятся на достаточном расстоянии друг от друга, а количество испускаемого света достаточно для корректной работы.

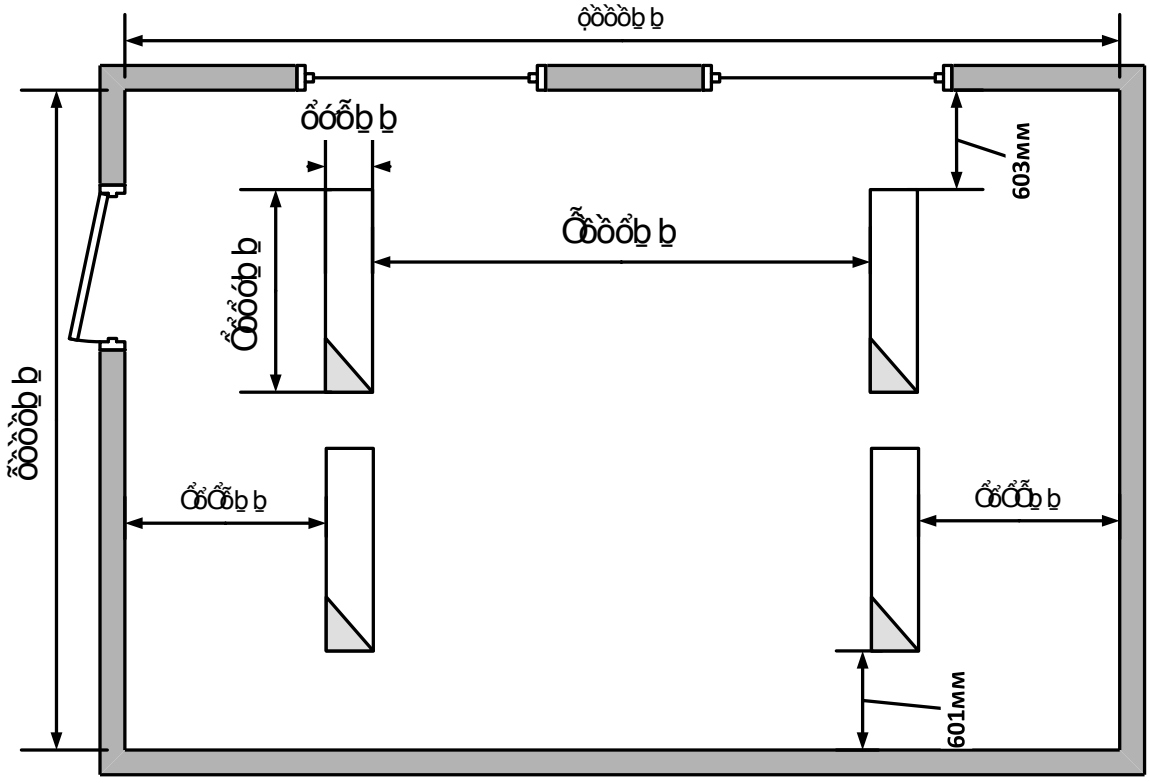

Рисунок 16 – Расположение ламп в помещении

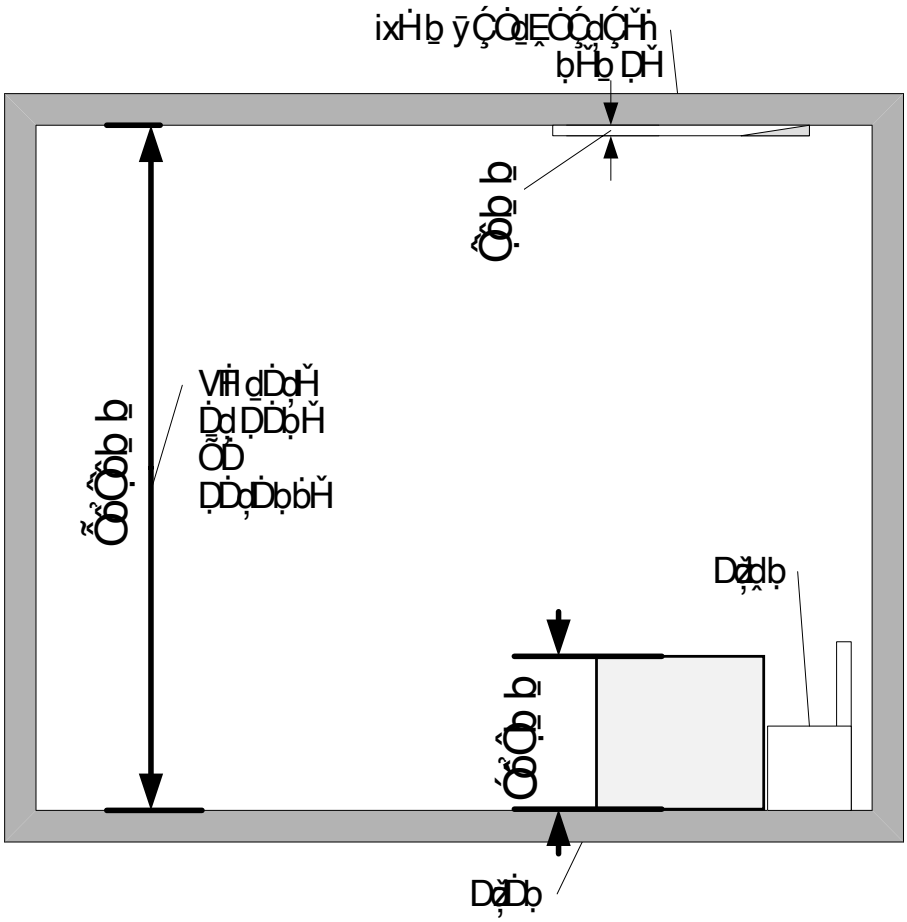

Рисунок 17 – Вид рабочего места сбоку

Согласно требованиям в ГОСТ 12.1.003-83 «Система стандартов безопасности труда (ССБТ). Шум. Общие требования безопасности», в помещениях с ЭВМ, где работают инженерно-технические работники, уровень шума не должен превышать 60 дБ·А.

В ПК источниками шума являются вентиляторы, трансформаторы, винчестеры, DVD-ROM. Для уменьшения шумов необходимо применять специально разработанные корпуса ПК, выполняющиеся в виде активного элемента охлаждающей системы. Все вентиляторы в системном блоке заменяются на радиаторы с теплоотводящими трубками. Все трубки имеют тепловые интерфейсы с корпусом, что позволяет удалять тепло от источников. Данный корпус не имеет ни одного вентилятора, т.е. шум, производимый работой ПК, сведен практически к нулю.

Использование лазерного принтера или размещение принтера вне кабинета позволит устранить излишние шумы от принтера.

Шум, создаваемый в люминесцентных лампах нового образца, почти сведен к нулю.

Оператор ПЭВМ подвергается электромагнитному излучению компьютера, результат которого зависит от напряженностей электрического и магнитного полей, индивидуальных особенностей организма и времени воздействия. Наиболее интенсивно электромагнитные поля воздействуют на органы с большим содержанием воды или со слабо развитой сосудистой системой.

Источником излучения электромагнитных волн в ЭВМ является электронно-лучевая трубка (ЭЛТ) дисплея и трансформаторы блоков питания.

В современных мониторах электромагнитное излучение невелико благодаря использованию защитных экранов. Для сведения излучения к нулю рекомендуется использовать жидкокристаллический монитор, поскольку его излучение значительно меньше.

Пользователя следует удалить от монитора на расстоянии не менее 500 мм (оптимальное расстояние - 600-700 мм).

5.2 Экологичность

Все устройства и техника, используемые при работе с программных обеспечением, со временем устаревает и изнашивается, появляется надобность в **утилизации отходов**. Под отходами понимают вещества или предметы, которые образованы в процессе производства, выполнения работ, оказания услуг или в процессе потребления, которые удаляются, предназначены для удаления или подлежат удалению, а утилизацией их – это повторное применение по прямому назначению (рециклинг), возврат их в производственный цикл после регенерации и извлечение полезных компонентов для повторного применения (рекуперация). Согласно ФЗ №89-ФЗ «Об отходах производства и потребления», классы отходов, которые зависят от степени негативного воздействия на окружающую среду, бывают следующими:

I класс – чрезвычайно опасные отходы;

II класс – высокоопасные отходы;

III класс – умеренно опасные отходы

IV класс – малоопасные отходы;

V класс – практически неопасные отходы.

В связи с этими классами, юридические лица или индивидуальные предприниматели обязаны обращаться к специализирующим фирмам, которые имеют лицензию на деятельность по сбору, транспортированию, обработке, утилизации, обезвреживанию, размещению отходов разных классов.

Это играет не маловажную роль ввиду использования различных металлов, включая драгоценных, которые используются потом повторно. Переработка отходов решает следующие задачи:

- экономическую – экономия ресурсов и контроль оборотов ресурсов в стране;

- экологическую – защита окружающей среды от пагубного воздействия.

Списывать технику, несмотря на дешевизну и простоту, не стоит по ряду причин:

- составляющие электронно-вычислительной техники относятся к отходам первого класса опасности из-за наличия разных тяжёлых химический соединений: ртуть, кадмий, мышьяк, свинец, цинк, никель и другие;

- юридические лица и индивидуальные предприниматели должны вести учёт драгоценных металлов, содержащиеся в основных средствах;

- за нарушение избавления отходов предусмотрены штрафные санкции.

Поэтому рекомендуется воспользоваться услугами лицензированной для утилизации отходов организации, с которой составляется договор о предоставлении экологических услуг по изъятию отходов.

Порядок утилизации состоит из следующего:

Порядок утилизации компьютеров:

- первый шаг – создание комиссии на предприятии, имеющем технику, подлежащую утилизации. Это внутренняя комиссия, которая создается для коллективного принятия решения о том, какая именно техника может быть списана;

- составление экспертного заключения о том, что техника действительно «отжила свое» и должна быть списана. В качестве эксперта может выступать как независимый специалист, так и сотрудник компании, имеющий диплом, подтверждающий его компетентность в работе с данной техникой;

- составление акта технической экспертизы, подтверждающего, что техника уже вышла из строя и не подлежит ремонту либо же что ремонт её уже нецелесообразен;

- составление акта списания компьютерной техники с обязательным отображением в бухгалтерском учете предприятия;

- утилизация техники на соответствующем предприятии, имеющем право на переработку компьютеров;

- получение официального подтверждения в виде документа, сообщающего о том, что техника была утилизирована в соответствующем порядке и опасные отходы не будут загрязнять окружающую среду.

5.3 Чрезвычайные ситуации

Одними из чрезвычайных ситуаций, возникающие на предприятии:

- пожары, взрывы, угроза взрывов – самые распространённые ЧС в современном обществе;

- угрозы терроризма;

- наводнения, ураганы, торнадо, землетрясения;

- угрозы выброса химических веществ на близлежащих промышленных предприятиях.

Федеральный закон от 21.12.1994 № 68-ФЗ «О защите населения и территорий от чрезвычайных ситуаций природного и техногенного характера», который в настоящий момент действует в редакции от 23.06.2016, обязывает организации к следующему:

- планировать и принимать меры по защите своих сотрудников от чрезвычайных ситуаций;

- планировать и устраивать мероприятия для повышения устойчивости работы своей организации в случае ЧС;

- обеспечивать создание и поддержание в полной готовности сил и средств предупреждения и ликвидации чрезвычайной ситуации, а также теоретическую и практическую подготовку своих сотрудников (инструктажи, специальные курсы, учения и тренировочные мероприятия);

- создавать и поддерживать в постоянной готовности системы оповещения о ЧС;

- руководствуясь планами действий по предупреждению и ликвидации последствий, обеспечивать проведение аварийно-спасательных и других работ на предприятии;

- финансировать мероприятия по защите своих работников от чрезвычайных ситуаций;

- обеспечить создание финансовых и материальных резервов для ликвидации последствий;

- обеспечивать своих работников информацией о защите населения и территорий от ЧС, а также оповещать об угрозе их возникновения или о возникновении; оказывать содействие федеральным органам, осуществляющим защиту

населения и территорий от ЧС, в деле установки средств оповещения, распространения информации и так далее.

При угрозе возникновения или возникновении чрезвычайной ситуации на территории организации ее руководителю вменяется в обязанность:

- ввести режим повышенной готовности;

- проинформировать о ситуации органы управления и силы единой государственной системы предупреждения и ликвидации последствий;

- принять решение об установлении уровня реагирования и других мер с целью защиты от чрезвычайной ситуации своих сотрудников и других лиц на территории организации.

5.4 Физические упражнения для глаз

Упражнение для снятие болезненных ощущений – зуда, жжения, сухости – и восстановления увлажняющего слоя глаз:

- в положении сидя или стоя, закрыть глаза и расслабить мышцы лба;

- медленно перевести глазные яблоки в крайнее левое положение, почувствовать напряжение глазных мышцы. Зафиксировать положение;

- медленно с напряжением перевести глаза вправо;

- повторить 9 раз.

При выполнении стараться не щуриться. Следить за тем, чтобы веки не подрагивали. Напряжение глазных мышц не должно быть чрезмерным.

Упражнение для восстановления гибкости экстраокулярных мышц и улучшения способности глаза к рефокусировке:

- прикрепить к стене бумажный крест;

- встать или сесть на расстояние 2,5-3 м от него;

- взять в руки карандаш и зафиксировать на нём взгляд, переводить взгляд на крест и обратно;

- повторить несколько раз.

ЗАКЛЮЧЕНИЕ

В век информации и развитых коммуникаций, в частности 3G и 4G, сеть Интернет проходит этап масштабной экспансии. Это напрямую повлияло на жизнь человека – передача различного рода информации стала важным фактором для качественной и даже эволюционной перемены. В связи с ежегодно растущим числом пользователей, имеющих доступ к сети Интернет, приобретает популярность разработка иного вида программного обеспечения – разработка веб-сайтов и веб-приложений.

Разработка подобных программных продуктов стала возможна ввиду совершенствования веб-технологий и веб-браузеров, позволяя воссоздавать функционал и интерфейс схожий с тем, что можно встретить в продуктах для настольных операционных систем. Такие приложения получили название одностраничных приложений, которые работают на стороне клиента в окне веб-браузера, архитектура которых является «клиент-сервер». Говоря о серверной части, то главная его цель – это хранение, обработка и выдачи пользовательских данных вкупе с предоставлением файлов приложения.

При выполнении бакалаврской работы был проведён анализ предметной области; проделано проектирование веб-приложения, определены требования для программного обеспечения; обоснован выбор языка программирования, среды разработки и использованных библиотек; описана база данных и представлены тексты запросов; изучены модели угрозы, нарушителя и разработаны меры предостережения, а также установлены рекомендации по безопасности и экологичности.

В качестве средств реализации компоненты были выбраны:

- среда разработки Visual Studio Code;
- язык программирования JavaScript;
- для модуля приложения библиотека ReactJS;
- для серверного модуля NodeJS.

Разработанное веб-приложение позволяет создавать и редактировать растровые изображения.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Боресков, А. В. Компьютерная графика : учебник и практикум для прикладного бакалавриата / А. В. Боресков, Е. В. Шикин. — Москва : Издательство Юрайт, 2018. — 219 с. — (Бакалавр. Прикладной курс). — ISBN 978-5-9916-5468-5. — Текст : электронный // ЭБС Юрайт [сайт]. — URL : https://biblio-online.ru/bcode/413327 (дата обращения: 20.05.2019).

2 Тузовский, А. Ф. Проектирование и разработка web-приложений : учебное пособие для академического бакалавриата / А. Ф. Тузовский. — Москва : Издательство Юрайт, 2018. — 218 с. — (Университеты России). — ISBN 978-5-534-00515-8. — Текст : электронный // ЭБС Юрайт [сайт]. — URL : https://biblio-online.ru/bcode/413954 (дата обращения: 20.05.2019).

3 Тузовский, А. Ф. Объектно-ориентированное программирование : учебное пособие для прикладного бакалавриата / А. Ф. Тузовский. — Москва : Издательство Юрайт, 2019. — 206 с. — (Университеты России). — ISBN 978-5-534-00849-4. — Текст : электронный // ЭБС Юрайт [сайт]. — URL : https://biblio-online.ru/bcode/434045 (дата обращения: 31.05.2019).

4 Шумилин, В. К. Типовая инструкция по охране труда для пользователей персональными электронно-вычислительными машинами (ПЭВМ) в электроэнергетике/ В.К. Шумилин – М. : Министерство энергетики Российской Федерации РАО «ЕЭС РОССИИ», 2001 – 89 с.

5 Эргономика // История компьютера URL : http://chernykh.net/content/view/164/ (дата обращения: 28.05.2019).

6 Digital 2019: Global Internet Use Accelerates // We are social URL : https://wearesocial.com/blog/2019/01/digital-2019-global-internet-use-accelerates (дата обращения: 26.05.2019).

7 Guides // NodeJS URL : https://nodejs.org/en/docs/guides/ (дата обращения: 18.05.2019).

8 Introducing JSX // React URL : https://reactjs.org/docs/introducing-jsx.html (дата обращения: 16.05.2019).

9 JavaScript // MDN URL : https://developer.mozilla.org/ru/docs/Web/JavaScript (дата обращения: 17.05.2019).

10 Top 5 Desktop Browsers // StatCounter Global Stats URL : http://gs.statcounter.com/#desktop-browser-ww-monthly-201810-201810-bar (дата обращения: 14.05.2019).

ПРИЛОЖЕНИЕ А

Программный продукт.

Исходный код файла index.js в серверном модуле.

```
import 'dotenv/config';
import cors from 'cors';
import bodyParser from 'body-parser';
import express from 'express';
import helmet from 'helmet'
import models, { connectDb } from './models';
import routes from './routes/routes';
const app = express();
app.use(cors());
app.use(helmet());
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: true }));
app.use(async (req, res, next) => {
  req.context = { models };
  next();
});
// Пути
app.use('/history', routes.history);
app.use('/users', routes.user);
// Запуск
connectDb().then(async () => {
  app.listen(process.env.PORT, () =>
    console.log(`Server on port ${process.env.PORT}!`),
  );
});
```

Исходный код файла index.js в модуле приложения.

```
import React from 'react';
import ReactDOM from 'react-dom';
import App from './containers/App';
import * as serviceWorker from './serviceWorker';
import { LocaleProvider } from 'antd';
```

```
import { i18nClient } from './locale';
import ruRU from 'antd/lib/locale-provider/ru_RU';
import enUS from 'antd/lib/locale-provider/en_US';
const antResources = {
   ru: ruRU,
   'ru-RU': ruRU,
   en: enUS,
   'en-US': enUS,
};
ReactDOM.render(
<LocaleProvider locale={antResources[i18nClient.language]}>
      <App />
</LocaleProvider>,
document.getElementById('root'));
serviceWorker.register();
```

Исходный код файла App.js.

```
import React, { Component } from 'react';
import '../styles/app-login.css';
import apiClass from '../components/Api';
import Login from '../components/Login';
import User from '../components/User';
import { deviceDetect } from 'react-device-detect';
import { Helmet } from 'react-helmet';
import ImageEditor from '../components/editor/ImageEditor';
import i18n from 'i18next';
class App extends Component {
  constructor(props) {
    super(props);
    this.state = {
      user: null,
      username: '',
      email: '',
      password: '',
      isLogged: false,
```

```
    text: ''
  };
}
handleLogin = async () => {
  let { email, password } = this.state;
  let user;
  if (email && password) {
    user = await apiClass.getUser(email, password, deviceDetect());
  }
  if (user.err) {
    this.setState({
      text: user.err
    });
    setTimeout(() => {
      this.setState({
        text: ''
      });
    }, 5000);
    return;
  }
  if (user) {
    await apiClass
      .getHistory(user._id)
      .then(data => {
        user.data = data;
        this.setState({
          user: user,
          isLogged: true
        });
      })
      .catch(err => console.log(err));
  }
};
handleLogout = async () => {
```

```
  await this.setState({
    username: '',
    email: '',
    password: '',
    isLogged: false,
    text: 'Вы успешно вышли.'
  });
};
handleEmail = e => {
  this.setState({
    email: e.target.value
  });
};
handleUsername = e => {
  this.setState({
    username: e.target.value
  });
};
handlePassword = e => {
  this.setState({
    password: e.target.value
  });
};
handleReset = async () => {
  return;
};
setPassword = value => {
  this.setState({
    password: value
  });
};
setText = value => {
  this.setState({
    text: value
```

```
  });
};
handleCreate = async () => {
  let { email, username, password } = this.state;
  if (email === '' || username === '' || password === '') {
    this.setState({
      text: 'Введите данные'
    });
    setTimeout(
      () =>
        this.setState({
          text: ''
        }),
      5000
    );
    return;
  }
  await apiClass
    .createUser(username, email, password, deviceDetect())
    .then(res => {
      if (res.status !== 403) {
        this.setState({
          user: res,
          isLogged: true
        });
      } else {
        this.setState({
          text: 'Пользователь зарегестрирован. Воспользуйтесь входом.'
        });
        setTimeout(
          () =>
            this.setState({
              text: ''
            }),
```

```
          5000
        );
      }
    })
    .catch(() => {
      this.setState({
        text: 'Ошибка.'
      });
      setTimeout(
        () =>
          this.setState({
            text: ''
          }),
        5000
      );
    });
};
render() {
  const { isLogged, user } = this.state;
  const helmet = (
    <Helmet>
      <meta charSet="utf-8" />
      <meta name="viewport" content="width=device-width, initial-scale=1.0" />
      <meta
        name="description"
        content="Редактор"
      />
      <link rel="manifest" href="./manifest.json" />
      <link rel="shortcut icon" href="./favicon.ico" />
      <title>{i18n.t('editor.title')}</title>
    </Helmet>
  );
  const renderApp =
    isLogged === true ? (
```

```
        <ImageEditor user={user} handleLogout={this.handleLogout} />
      ) : (
        <Login
          username={this.state.username}
          handleUsername={this.handleUsername}
          password={this.state.password}
          handlePassword={this.handlePassword}
          email={this.state.email}
          handleEmail={this.handleEmail}
          text={this.state.text}
          handleLogin={this.handleLogin}
          handleCreate={this.handleCreate}
          handleReset={this.handleReset}
          setPassword={this.setPassword}
          setText={this.setText}
        />
      );
    return (
      <div className="rde-main">
        {helmet}
        <div className="rde-content">{renderApp}</div>
      </div>
    );
  }
}
export default App;
```