**Министерство науки и высшего образования Российской Федерации**

**федеральное государственное автономное образовательное учреждение высшего образования**

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМТО»**

Факультет СПО

Направление подготовки (специальность) 09.02.07 Информационные системы и программирование

**О Т Ч Е Т**

**о преддипломной практике**

Тема задания: Разработка прототипа клиентской части автоматизированной информационной системы для клиники «Стоматология Доктора Вознесенского» \_

Обучающийся Кунаев Н.Д. Группа Y2432

*(Фамилия И.О.) (номер группы)*

Согласовано:

|  |  |
| --- | --- |
| Руководитель практики от профильной организации: | Костяев Александр Александрович, заместитель генерального директора ООО «АКАДЕМИЯ ЦИФРОВОЙ КУЛЬТУРЫ» |
|  | (Фамилия И.О., должность и место работы) |
| Руководитель практики от Университета ИТМО: | Говорова Марина Михайловна, преподаватель факультета СПО |
|  | (Фамилия И.О., должность и место работы) |

Практика пройдена с оценкой *«отлично»*

«19» мая 2021г.

Санкт-Петербург

2021

**Министерство науки и высшего образования Российской Федерации**

**федеральное государственное автономное образовательное учреждение высшего образования**

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Факультет СПО**

**И Н Д И В И Д У А Л Ь Н О Е З А Д А Н И Е**

**на преддипломную практику**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Студент** | Кунаев Н.Д. | | | **Группа №** | **Y2432** |
| (Фамилия И. О.) | | | | | |
| **Руководитель от профильной организации** | | | Костяев Александр Александрович, заместитель генерального директора ООО «АКАДЕМИЯ ЦИФРОВОЙ КУЛЬТУРЫ» | | |
| (Фамилия И. О., место работы, должность) | | | | | |
| **Руководитель от Университета ИТМО** | | Говорова Марина Михайловна, преподаватель факультета СПО | | | |
| (Фамилия И. О., место работы, должность) | | | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Тема задания:** | Разработка клиентской части автоматизированной информационной | | |
| системы для клиники «Стоматология Доктора Вознесенского» | | | |
| **Сроки прохождения практики:** | | 22.04.2021 – 19.05.2021 | |
| **Место прохождения практики:** | | ООО «АКАДЕМИЯ ЦИФРОВОЙ КУЛЬТУРЫ» | |
| **Должность практиканта:** | | - | |
| **1. Виды работ и требования к их выполнению:** | |  | |
| Производственная практика проводится непрерывно на базе ООО «АКАДЕМИЯ ЦИФРОВОЙ КУЛЬТУРЫ»***.*** В ходе прохождения практики выполняются следующие виды работ: | | | |
| 1. Вводный инструктаж по технике безопасности и общим целям, и задачам практики. | | | |
| 1. Анализ поставленной задачи: уточнение требований технического задания. | | | |
| 1. Детальное проектирование системы/приложения/программы. | | | |
| 1. Реализация, отладка и тестирование системы/приложения/программы. | | | |
| 1. Формирование отчетных материалов и представление программы. | | | |
|  | | | |
| **2. Виды отчетных материалов и требования к их оформлению:** | | |  |
| По результатам прохождения практики составляется отчет, в котором представляется анализ поставленной задачи, описание процесса проектирования системы/приложения/программы, модель системы/приложения/программы и техническое задание. Программа представляется руководителю(ям) практики. Оформление отчета должно соответствовать Рекомендациям по оформлению технических документов факультета СПО Университета ИТМО. | | | |

**3. ПЛАН-ГРАФИК**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| №  этапа | Наименование этапа | Срок завершения этапа | Виды работ | Форма отчетности |
|  | Вводный инструктаж | 22.04.2021 – 25.04.2021 | Ознакомление с инструкцией по технике безопасности. Ознакомление с целями и задачами производственной практики | Журнал по технике безопасности |
|  | Анализ поставленной задачи | 26.04.2021 – 02.05.2021 | Анализ технического задания. Уточнение требований (при необходимости). | Отчет по практике.  Дневник прохождения практики. |
|  | Детальное проектирование системы/приложения/программы. | 03.05.2021 – 16.05.2021 | Детальное проектирование модулей (составных частей программы). | Отчет по практике.  Дневник прохождения практики. |
|  | Реализация, отладка и тестирование системы/приложения/программы. | Реализация, отладка и тестирование системы/приложения/программы (доработка прототипа). | Отчет по практике.  Дневник прохождения практики. |
|  | Подготовка отчетных материалов и представление программы | 17.05.2021 – 19.05.2021 | Формирование отчета о прохождении практики, демонстрация работы системы/приложения/программы. | Отчет по практике.  Дневник прохождения практики. |

**4.  ТРЕБОВАНИЯ К ОФОРМЛЕНИЮ ОТЧЕТНОСТИ**

|  |  |
| --- | --- |
| Наименование формы отчетности | Требования к оформлению |
| Отчет о практике | Отчет о преддипломной практике оформляется в соответствии с Методическими указаниями по преддипломной практике по специальности 09.02.07 Информационные системы и программирование и Рекомендациям по оформлению технических документов факультета СПО Университета ИТМО. Титульный лист отчета оформляется в соответствии с Приложением В Методических указаний. |
| Индивидуальное задание | Индивидуальное задание оформляется в соответствии с Методическими указаниями по преддипломной практике по специальности 09.02.07 Информационные системы и программирование (Приложение А) и включается в отчет о преддипломной практике после титульного листа. |
| Дневник практики | Дневник практикиоформляется в соответствии с Методическими указаниями по преддипломной практике по специальности 09.02.07 Информационные системы и программирование (Приложение Б). |
| Аттестационный лист | Аттестационный лист оформляется в соответствии с Методическими указаниями по преддипломной практике по специальности 09.02.07 Информационные системы и программирование (Приложение Г) |

**Задание выдано:**

**Руководитель практики от Университета ИТМО\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** / М. М. Говорова

(подпись руководителя)                                     ФИО

«22» апреля 2021г.

Согласовано:

**Руководитель практики от профильной организации \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** / А. А. Костяев

(подпись руководителя)                           ФИО

«22» апреля 2021г.

**Задание принял к исполнению:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** / Н. Д. Кунаев

 (подпись обучающегося)                 ФИО

ВВЕДЕНИЕ

Целью преддипломной практики является разработка веб-приложения для медицинской клиники «Стоматология Доктора Вознесенского» Фрунзенского района Санкт-Петербурга. Практика проходит на базе ООО «Академия цифровой культуры».

Для достижения цели, поставленной в рамках преддипломной практики, необходимо выполнить следующие задачи:

1. Анализ технического задания;
2. Уточнение функциональных требований (при необходимости);
3. Программная реализация системы (клиентской части);
4. Проведение отладки и тестирования системы;
5. Формирование технического описания разработанной программной системы;
6. Представление реализованной программной системы.

Система позволяет хранить информацию о пациентах, о предоставленных услугах и о расписании работы врачей.

Отчёт содержит следующие разделы:

1. Введение;
2. Раздел 1 Общие сведения о разработке;
3. Раздел 2 Функциональное назначение;
4. Раздел 3 Описание логической структуры;
5. Раздел 4 Сведения о входных и выходных данных;
6. Раздел 5 Используемые технические средства;
7. Раздел 6 Вызов и загрузка программы;
8. Заключение;
9. Список источников;
10. Приложение А Техническое задание;
11. Приложение Б Модель системы.
12. ОБЩИЕ СВЕДЕНИЯ О РАЗРАБОТКЕ
    1. Наименование программы

АИС для медицинской клиники «Стоматология Доктора Вознесенского» (клиентская часть).

* 1. Программное обеспечение, необходимое для функционирования программы

Для функционирования программы необходимы следующие веб-браузеры:

* Google Chrome, версии 90.0.4430.212+;
* Mozilla Firefox Developer Edition, версии 89.0 x64+;
* Microsoft Edge, версии 90.0.818.56+;
* Safari, версии 14.1+;
* Opera, версии 76.0.4017.123+.
  1. Программные средства

В приложении использованы парадигмы реактивного, декларативного, объектно(прототипно)-ориентированного и функционального программирования.

Для реализации были использованы:

Во время разработки клиентской части:

NodeJs Server, версии 14.15.4;

Пакетный менеджер Yarn, версии 1.22.10;

Метаязык SASS, версии 1.32.5;

ReactJS, версии 17.0.1;

Bootstrap, версии 4.6.0;

React-bootstrap, версии 1.5.0.

В продакшн версии веб-приложения:

HTML, версии 5.3;

CSS, версии 2.1;

JS, версии ES9 (ECMAScript 2018).

1. Функциональное назначение
   1. Описание решаемых задач

Система предоставляет различный функционал для разных групп пользователей (директор/администратор/врач).

Врач может просматривать своё расписание, просматривать и вносить изменения в медицинские карты пациентов.

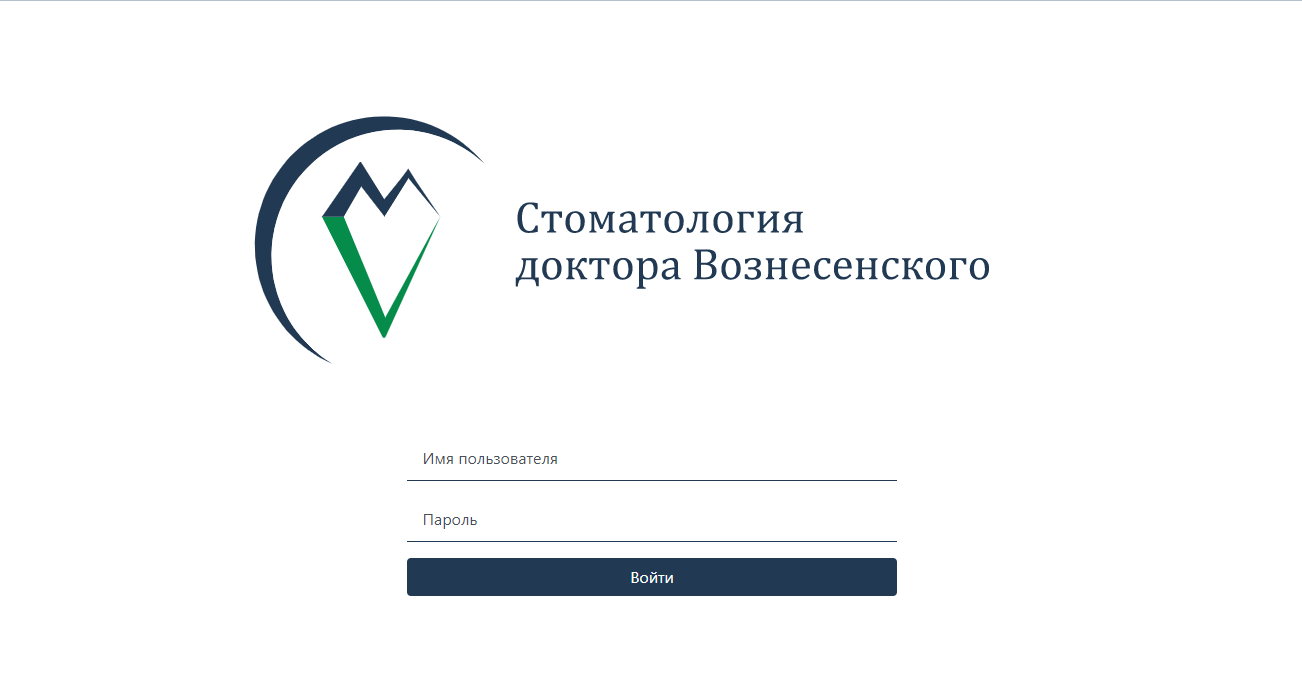
Администратор может делать то, что может делать врач, а также вносить изменения в расписание, добавлять новых пациентов и печатать накладные и прочие документы по выполненным услугам, вписывать пациентов в расписание врача (записывать на приём).

Директору доступен весь функционал других категорий пользователей, а также он может добавлять новые услуги, изменять их цену, добавлять новых пользователей, изменять любые данные пользователей, назначать рабочие дни для администраторов и врачей, добавлять новые товары и изменять их цены.

Просмотр расписания реализован посредством таблицы, отображающей календарь с метками врачей, работающих в определённые дни.

* 1. Описание функций

При входе в веб-приложение открывается форма авторизации (рисунок 1). После ввода корректных данных (логин и пароль) и нажатия на кнопку «Войти», исходя из введенных данных, определяется к какой группе прав относится пользователь, в зависимости от неё пользователь будет видеть разные варианты интерфейса (пока реализован только интерфейс директора). Форма входа показана на рисунке 1.



*Рисунок 1 – Окно аутентификации пользователей*

После удачной аутентификации происходит переадресация на главную страницу, на которой можно увидеть расписание в виде календаря, а так меню навигации, позволяющее перейти в другие разделы веб-приложения.

На главной странице пока ничего нет, кроме боковой панели навигации.



*Рисунок 2 – Окно аутентификации пользователей*

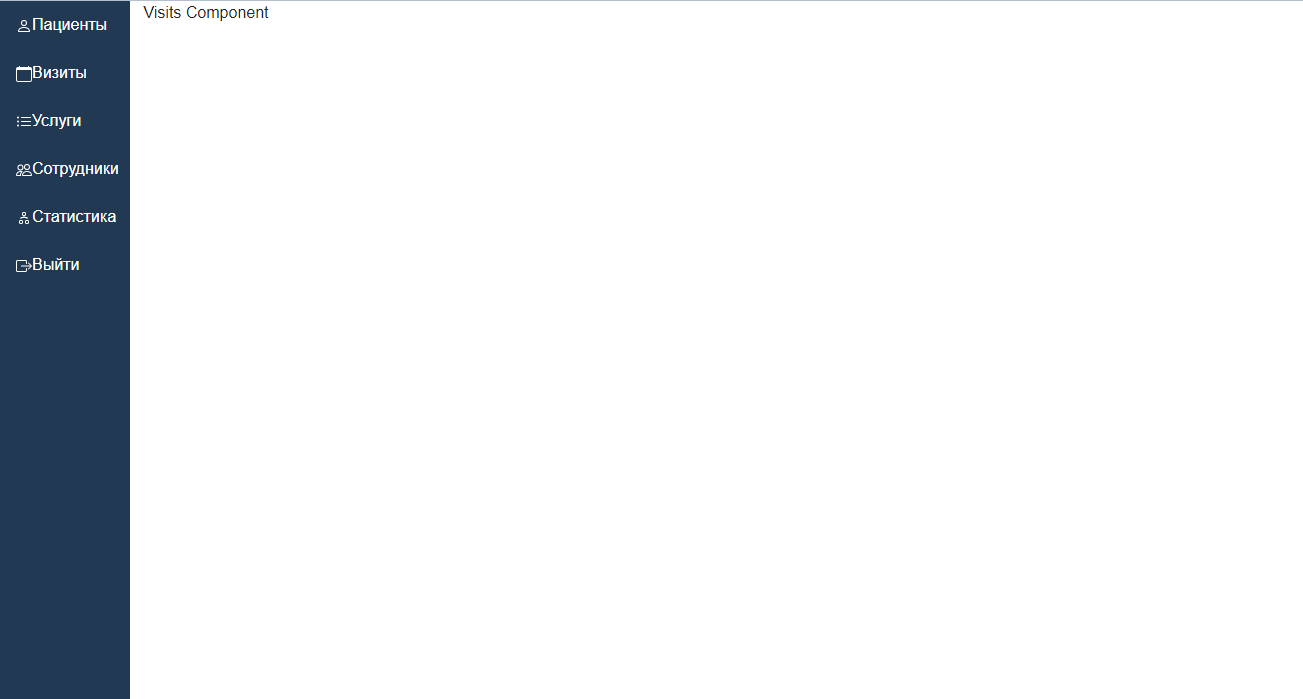
Перейдя по ссылке «Пациенты», открывается окно со списком всех пациентов, которое показано на рисунке 3.

Изображение выглядит как стол

Автоматически созданное описание

*Рисунок 3 – Окно «Пациенты»*

При переходе по ссылкам «Визиты» и «Услуги» выводятся пустые страницы с ещё нереализованным функционалом, которые показаны на рисунках 4 и 5.



*Рисунок 4 – Окно «Визиты»*



*Рисунок 5 – Окно «Услуги»*

При переходе по ссылке «Сотрудники» открывается список всех зарегистрированных сотрудников.

Изображение выглядит как стол

Автоматически созданное описание

*Рисунок 6 – Окно «Сотрудники»*

При переходе по ссылке «Статистика» открывается окно с базовой статистикой, которая представлена на рисунке 6.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 7 – Окно «Статистика»*

* 1. Средства контроля правильности выполнения

Поскольку доступ к веб-приложению есть только посредствам локальной сети, то снижается общий уровень уязвимости системы и хранящихся данных.

* 1. Ограничения области применения программы

Программа может быть использована только в медицинской клинике «Стоматология Доктора Вознесенского» (Санкт-Петербург, Фрунзенский район).

1. Описание логической структуры
   1. Алгоритм программы

Функциональная модель представлена в BPMN-диаграмме, на рисунке Б.1, включающей основной бизнес-процесс, который описывает процесс записи пациентов на приём, который включает в себя проведение лечения и оплаты. Описание поведения пользователей представлено на рисунке Б.2 в формате «Пользовательская история» (User story).

* 1. Используемые методы при разработке программы

В процессе разработки системы были использованы:

* git – для контроля версий файлов, версии 2.29.2.windows.3;
* https – протокол для обмена данными между клиентом и сервером;
* json – универсальный формат данных;
* yarn – пакетный менеджер, версии 1.22.10;
* sass – мета-язык для каскадных таблиц стилей (css), версии 1.32.5;
* html – стандартизированный язык разметки веб-документов, версии 5.3;
* ESLint – линтер для JavaScript, статический анализатор кода, версии 0.2.1.

При разработке были использованы следующие паттерны проектирования:

* Простые компоненты (stateless component);
* Условный рендеринг (conditional rendering);
* Типы потомков (children types);
* Массив как потомок (array as children);
* Функция как потомок (function as children);
* Функция в render (render callback);
* Перенаправление компонента (proxy component);
* Компоненты высшего порядка (higher-order component);
* Передача свойств вниз по дереву компонентов (passing down props);

После разработки производится сборка и компиляция dev-проекта в продакшн билд, который более оптимизирован для браузеров и работы в интернете: все js- и css-/sass-файлы собираются в набор минимизированных файлов, чтобы браузер загружал сразу весь функционал и не осталось не подгруженных файлов.

* 1. Структура программы

В таблице 1 приведена структура клиентской части веб-приложения до продакшн сборки.

Таблица 1 – Структура клиентской части приложения

|  |  |  |
| --- | --- | --- |
| Файл | Функции | Назначение |
| 1 | 3 | 2 |
| Auth.jsx | Auth | Функция, возвращающая визуальный компонент окна авторизации |
| Auth.setField | Функция, для логирования изменений поля и записи данных в структуру |
| Auth.setAuthorization | Функция, устанавливающая флаг авторизации |
| Auth.module.sass | - | Файл стилей |
| SideBar.jsx | SideBar | Возвращает визуальный компонент бокового меню |
| SideBar.module.sass | - | Файл стилей |

Продолжение таблицы 1

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Client.jsx | Client | Функция, возвращающая визуальный интерактивный компонент с информацией о пациенте |
| Client.archiveClient | Функция архивации клиентов |
| Client.dearchiveClient | Функция разархивации клиентов |
| ClientList.jsx | ClientList.componentDidMount | Функция, запускающаяся в момент загрузки компонента Client, которая вызывает функцию getClietns |
| ClientList.getClients | Функция, получающая данные пользователей из Базы Данных по средствам API |
| ClietnList.render | Функция, возвращающая разметку списка клиентов и передающая данные в класс Client |
| ClientList.module.sass | - | Файл стилей |
| ClientAddModalWrapper.jsx | ClientAddModalWrapper | Функция-обёртка для модальных окон, добавляющая возможность обновления и добавления клиентов |
| ClientEditModal.jsx | ClientEditModal | Функция, которая возвращает модальное окно с формой изменения существующих пациентов |
| ClientEditModal.setField | Функция, для логирования изменений поля и записи данных в структуру |
| ClientEditModal.handleSubmit | Функция, собирающая данные и проверяющая на наличие ошибок |
| ClientEditModal.findFromErrors | Функция поиска ошибок во введённых данных |

Продолжение таблицы 1

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| ClientAddModalClass.jsx | ClientAddModal | Функция, которая возвращает модальное окно с формой добавление пациентов |
| ClientAddModal.setField | Функция для логирования изменений поля и записи данных в структуру |
| ClientAddModal.handleSubmit | Функция, собирающая данные и проверяющая на наличие ошибок |
| ClientAddModal.findFromErrors | Функция поиска ошибок во введённых данных |
| Employee.module.sass | - | Файл стилей |
| EmployeesList.jsx | Employee | Функция, возвращающая интерактивный визуальный компонент со списком сотрудников |
| EmployeesList.module.sass | - | Файл стилей |
| EmployeeAddModal.jsx | EmployeeAddModal | Функция, которая возвращает модальное окно с формой добавление сотрудников |
| EmployeeAddModal.setField | Функция, для логирования изменений поля и записи данных в структуру |
| ClientAddModal.handleSubmit | Функция, собирающая данные и проверяющая на наличие ошибок |
| ClientAddModal.findFromErrors | Функция поиска ошибок во введённых данных |
| EmployeeAddModalWrapper.jsx | EmployeeAddModalWrapper | Функция-обёртка для модальных окон, добавляющая возможность обновления и добавления сотрудников |

Продолжение таблицы 1

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| EmployeeEditModal.jsx | ClientEditModal | Функция, которая возвращает модальное окно с формой изменения существующих сотрудников |
| ClientEditModal.setField | Функция, для логирования изменений поля и записи данных в структуру |
| ClientEditModal.handleSubmit | Функция, собирающая данные и проверяющая на наличие ошибок |
| ClientEditModal.findFromErrors | Функция поиска ошибок во введённых данных |
| Employee.jsx | Employee | Функция, возвращающая интерактивный визуальный компонент интерфейса, которые выводит таблицу всех сотрудников |
| SideBar.module.sass | - | Файл стилей |
| SideBar.jsx | SideBar | Функция, возвращающая разметку меню боковой навигации |
| Services.module.sass | - | Файл стилей |
| Stats.jsx |  | Функция, возвращающая статистические данные |
| Visits.jsx | Visits | Функция, возвращающая список визитов |
| Visits.module.sass | - | Файл стилей |
| WorkingSpace.jsx | WorkingSpace | Функция-маршрутизатор, определяет, что должно быть выведено, в зависимости от адреса |
| WorkingSpace.module.sass | - | Файл стилей |
| Services.jsx | Service | Функция, возвращающая список всех предоставляемых услуг |
| Stats.module.sass | - | Файл стилей |
| Api.js | axios.create | Обёртка функции из библиотеки axios |
| App.js | App | Функция-маршрутизатор, которая не позволяет неавторизованным пользователям получать доступ к системе |

Продолжение таблицы 1

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| App.sass | - | Файл стилей |
| index.css | - | Файл стилей |
| index.js | ReactDom.render | Функция, запускающая всё веб-приложение |
| package.json |  | Файл, в котором перечислены зависимости |
| yarn.lock |  | Файл, в котором перечислены зафиксированные версии зависимостей |
| .gitignore |  | Файл, в котором перечислены игнорируемые файлы |
| public/index.html | - | Файл разметки стартового веб-документа |
| public/index.css | - | Файл стилей |

В таблице 2 представлена структура программы после продакшн сборки.

Таблица 2 – Структура клиентской части веб-приложения

|  |  |  |
| --- | --- | --- |
| Файл | Функции | Назначение |
| index.html | - | Файл разметки стартового веб-документа |
| index.css | - | Файл стилей |
| static/css/\*.\*.chunk.css | - | Минимизированные файлы стилей |
| static/js/\*.\*.chunk.js | Их слишком много, чтобы это было возможным описать | Минимизированные файлы js, содержащий основной функционал |
| static/js/\*.\*.chunk.js.LICENSE | - | Файлы лицензий |
| static/js/main.\*.chunk.js | Их слишком много, чтобы это было возможным описать | Минимизированный файл webpack-скрипта, содержащий фундаментальные настройки сборки окружения |
| static/js/runtime-main.\*.js | Их слишком много, чтобы это было возможным описать | Минимизированный файл js |

* 1. Форматы обмена данными

Основной формат обмена данными – обращение к Rest API в виде JSON-запросов.

1. Сведения о входных и выходных данных

Входные и выходные данные приложения описаны в приложения А «Техническое задание» в разделе 3.2.

1. Используемые технические средства

Для использования разрабатываемого веб-приложения устройство пользователя должно обладать следующими техническими требованиями, соответствующим требованиям использования основных браузеров (Chrome, Firefox, Internet Explorer, Opera, Safari и т.п.) (URL: <https://support.google.com/chrome/a/answer/7100626>, дата обращения: 18.05.21):

* ОС Windows 7 (поддержка завершится не ранее 15.01.2022), Windows 8, Windows 8.1, Windows 10 или более поздней версии или OS X El Capitan 10.11 или более поздней версии.
* процессор Intel Pentium 4 или более поздней версии с поддержкой SSE3.

1. Вызов и загрузка системы
   1. Конфигурирование и загрузка программы

Развертывание системы производится по следующему алгоритму:

Добавить конфигурацию из файла

1. Запуск сервера компонента логирования;
2. Запуск сервера компонента хранения;
3. Запуск компонента API;
4. Запуск веб-сервера.

Алгоритм также может быть описан следующей последовательностью команд (при использовании описанных в разделе 5 технических средств):

systemctl influxd start;

systemctl mysqld start;

java -jar ./voz-backend.jar;

systemctl nginx start.

* 1. Способ вызова и завершения работы программы

Разрабатываемая программа – веб-приложение, поэтому для начала работы требуется перейти на определённый URL, который будет назначен заказчиком на рабочем месте.

Для прекращения работы необходимо покинуть браузер или вкладку, в которой открыто веб-приложение.

* 1. Режимы работы программы

Пользователи взаимодействуют с веб-приложением в интерактивном режиме.

Заключение

В результате выполнения преддипломной практики было разработано веб-приложение для учёта пациентов в соответствии с техническим заданием и моделями разработки. Сформировано техническое описание разработанной системы.

Для достижения поставленной цели были выполнены следующее задачи:

1. Уточнение функциональных требований (при необходимости).
2. Программная реализация системы (клиентской части).
3. Проведение отладки и тестирования системы.
4. Формирование технического описания разработанной программной системы.
5. Представление реализованной программной системы.

Таким образом, все задачи решены в полном объёме. Цель практики достигнута.

В качестве перспективы дальнейшего развития приложения можно рассмотреть расширение системы для внедрения в другие частные медицинские клиники.

ПРИЛОЖЕНИЕ А  
Техническое задание

ТЕРМИНЫ, ПОНЯТИЯ И СОКРАЩЕНИЯ

В данном разделе представлены основные термины, понятия и сокращения, используемые в настоящем документе – смотреть [Таблица 1].

Таблица 1 – Термины, понятия и сокращения

|  |  |
| --- | --- |
| **Термин, понятие или сокращение** | **Определение** |
| БД | База данных |
| ОС | Операционная система |
| ПО | Программное обеспечение |
| Система | Разрабатываемый программный продукт |
| СУБД | Система управления базой данных |
| ТЗ | Техническое задание |
| GUI | Global User Interface – Глобальный пользовательский интерфейс |

1. НАЗНАЧЕНИЕ РАЗРАБОТКИ

Назначением разработки является создание веб-приложения для автоматизации учёта пациентов медицинской клиники «Стоматология Доктора Вознесенского».

Программа должна эксплуатироваться на объекте заказчика. Пользователями программы должны являться администраторы, директор и врачи. Пациенты не должны иметь доступ к системе.

1. ЦЕЛЬ И ЗАДАЧИ, РЕШАЕМЫЕ В ПРОЦЕССЕ ДОСТИЖЕНИЯ ЦЕЛИ

Целью является создание клиент-серверной автоматизированной информационной системы (АИС) для организации работы СК.

Задачи, решаемые в процессе достижения цели:

* уточнение требований к системе;
* детальное проектирование системы;
* программная реализация;
* тестирование.

1. ТРЕБОВАНИЯ К системе
   1. Требования к функциональным характеристикам

Система должна обеспечить хранение информации о клиентах, предоставленных услугах и графике работы сотрудников.

В системе должны быть определены следующие категории пользователей:

* администратор;
* директор;
* врач.

В системе должны быть реализованы следующие функциональные возможности:

* + - работа с расписаниями сотрудников, изменение графиков работы;
    - добавление, удаление и изменение информации о приемах пациентов;
    - учет оказываемых клиникой услуг, их классификация по номенклатуре медицинских услуг;
    - учет клиентов клиники, изменение и добавление информации о них, добавление клиента в архив и удаление из него;
    - учет дисконтных карт клиентов;
    - возможность ограничения прав.
  1. Описание входных и выходных данных

Входными данными приложения являются следующие данные.

О пациентах:

* паспортные данные;
* ФИО;
* дата рождения;
* адрес регистрации;
* адрес проживания;
* профессия;
* телефон;
* пол;
* место рождения;
* статус архивации;
* откуда узнал;
* адрес электронной почты;
* комментарий;
* баланс;
* постоянная скидка.

О визитах:

* пациент;
* дата;
* статус первичности;
* причина;
* посещённые врачи;
* комментарий;
* статус.

О сотрудниках:

* ФИО;
* Дата рождения;
* должность;
* контактные данные.
* логин;
* пароль;
* права доступа.

Об услугах:

* цена;
* наименование;
* группа.

О предоставленных услугах:

* услуга;
* фактическая цена;
* посещение, в которое услуга оказана.

О товарах:

* цена;
* наименование;
* группа;
* количество.

О проданных товарах:

* товар;
* клиент;
* фактическая цена;
* дата продажи;
* факт использования дисконтной карты;

О дисконтных картах:

* клиент;
* размер скидки.

О транзакциях:

* клиент;
* изменение баланса.
  1. Модель приложения

Функциональная модель поведения приложения должна соответствовать модели на рисунках Б.1.

Модель поведения пользователей должна соответствовать модели, представленной на рисунке Б.2.

Модель прикладного программного интерфейса должна соответствовать модели на рисунке Б.3.

* 1. Эргономические и технико-эстетические требования

При проектировании форм необходимо стремиться к использованию ограниченного набора цветов и уделять внимание их правильному сочетанию. По возможности должны использоваться системные цвета и свойственные системе элементы управления. Набор из трёх цветов продиктован основными цветами логотипа стоматологической клиники.

При неправильно веденных данных пользователю должно быть показано диалоговое окно с ошибкой (язык ошибки в данном случае может быть русским или английским).

Правила валидации полей определяются на стороне клиента.

* 1. Структура системы

Система должна включать в себя следующие компоненты клиентского интерфейса (подсистемы):

* пациенты – модуль, содержащий информацию о клиентах в системе;
* сотрудники – модуль, содержащий информацию о пользователях системы;
* визиты – модуль, содержащий информацию о посещениях пациентов клиники и посещённых врачах;
* услуги – модуль, содержащий информацию о возможных услугах, предоставляемых клиникой;
* статистика – модуль, содержащий статистическую информацию о работе клиники и о клиентах.
  1. Требования к информационному обеспечению системы

Для обеспечения целостности данных должны использоваться системы валидации форм. Доступ к данным должен быть предоставлен только авторизованным пользователям с учетом их прав доступа.

* 1. Требования к технологиям разработки

При разработке приложения должны использоваться парадигмы реактивного, декларативного, объектно(прототипно)-ориентированного и функционального программирования. Архитектура реализуемой системы – клиент-серверное веб-приложение.

* 1. Требования к программным средствам и технологиям разработки

Компонента клиентского интерфейса должно быть реализовано при помощи следующих средств:

* язык программирования JavaScript (версии ECMAScript 9 или выше);
* язык размети HTML5;
* язык стилей CSS3;
* сервер для разработки Node.Js версии не ниже 14.15.4;
* пакетный менеджер yarn версии не ниже 1.22.10;
* интегрированная среда разработки Web Storm версии 2020.3.3.

Также для реализации потребуются некоторые библиотеки и метаязыки:

* библиотека ReactJs;
* метаязык SASS;

При эксплуатации приложения должны быть установлены:

* любой современный браузер (Mozilla Firefox, Opera, Google Chrome, Safari, Yandex Browser).
  1. Требования к защите от несанкционированного доступа

Контроль защиты информации от несанкционированного доступа осуществляется на стороне клиента по средствам использования токена авторизации.

Компонент прикладного программного интерфейса должен обеспечивать защиту от несанкционированного доступа:

* аутентификация пользователя;
* проверку полномочий пользователя при работе с системой.

Защищённая часть системы должна использовать пароли, символы должны маскироваться одним типом символов (\*).

Директор имеет доступ ко всем функциям системы.

Администраторы не могут вносить изменения в прайс-листы, не может добавлять новых пользователей, не может изменять существующих пользователей.

Врачи могут получить доступ только к информации о пациентах, их визитах и выполненных услугах.

1. ТРЕБОВАНИЯ К МЕТОДАМ ТЕСТИРОВАНИЯ

Разрабатываемое веб-приложение будет тестироваться с помощью технологий белого и чёрного ящиков.

С помощью метода белого ящика будут проведены:

* модульное тестирование;
* интеграционное тестирование.

Тестирование веб-приложений проходит через несколько этапов [2]:

* на этапе функционального тестирования проверяются:
  + функциональные требования (с помощью метода чёрного ящика);
  + загрузка требуемых файлов (с помощью метода чёрного ящика);
* на этапе тестирования вёрстки проверяются:
  + соответствие вёрстки макету (чёрный ящик);
  + оптимизация графических изображений (чёрный ящик);
  + валидность кода [3] (белый ящик);
  + совместимость (чёрный ящик);
* на этапе тестирования удобства интерфейса:
  + собираются данные и отзывы пользователей (чёрный ящик);
  + анализ собранных данных и применение результатов анализа в последующей разработке;
* на этапе тестирования безопасности:
  + тестирование быстродействия сайта [4];
  + тестирование безопасности по OWASP testing guide [6];

ПРИЛОЖЕНИЕ Б

Модель системы

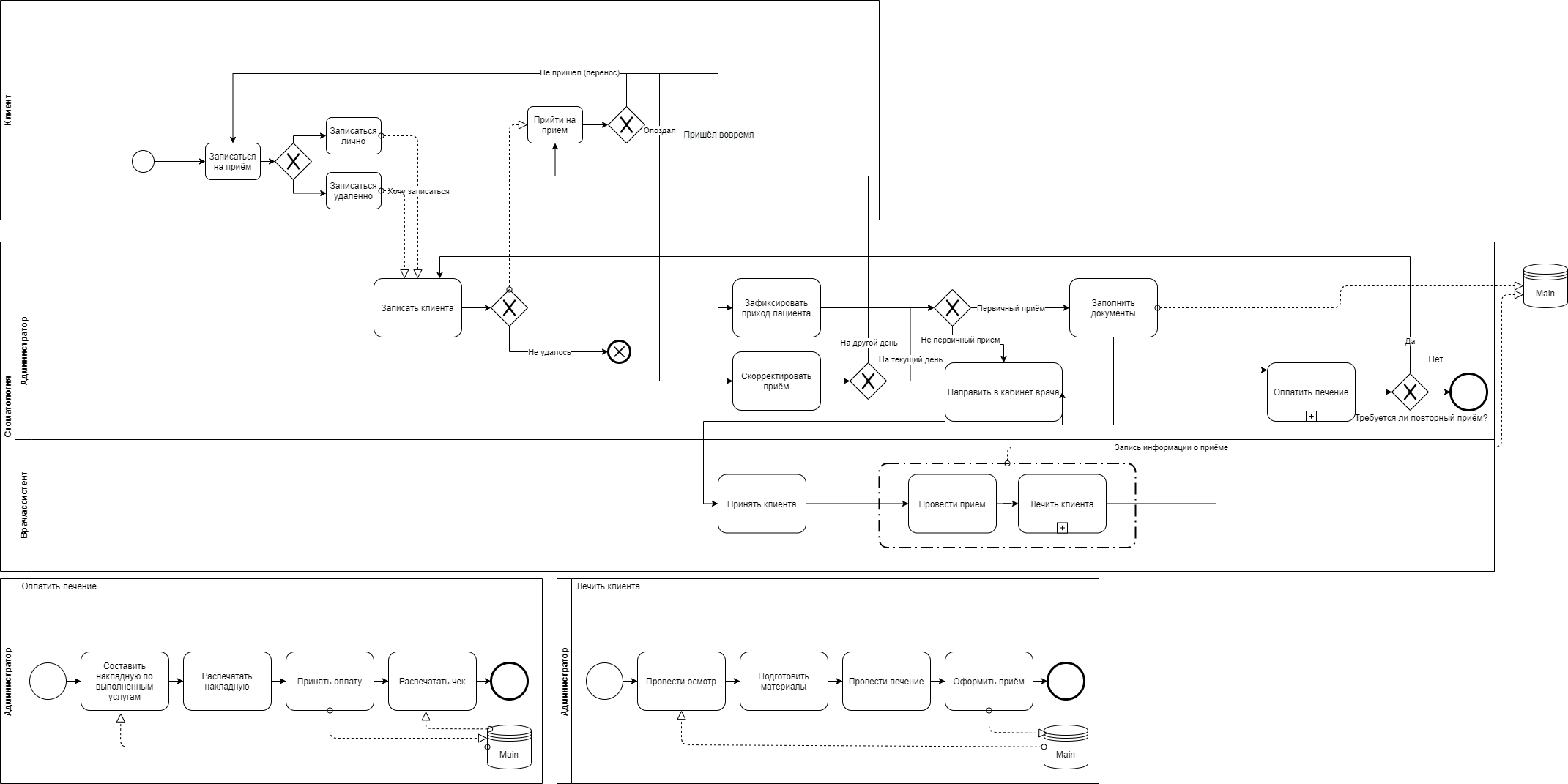
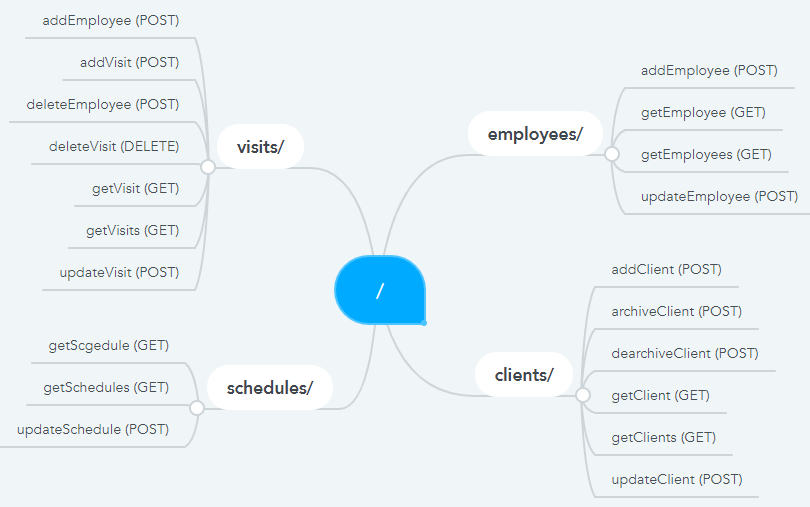


Рисунок Б.1 – Функциональная модель системы в нотации BPMN

Изображение выглядит как текст

Автоматически созданное описание

Рисунок Б.2 – Концептуальная модель: диагрмма User Story



*Рисунок Б.3 – Модель прикладного программного интерфейса*