

Министерство образования и науки Российской Федерации  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО  
ОБРАЗОВАНИЯ

**“САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ  
ИССЛЕДОВАТЕЛЬСКИЙ  
УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ,  
МЕХАНИКИ И ОПТИКИ”**

Факультет ИКТ

Образовательная программа Интеллектуальные системы в гуманитарной  
сфере

Направление подготовки (специальность) Интеллектуальные системы в  
гуманитарной сфере (45.03.04)

**О Т Ч Е Т**

по курсовой работе

Тема задания: РЕАЛИЗАЦИЯ WEB-СЕРВИСОВ СРЕДСТВАМИ Django REST framework.

Обучающийся: Козырева Алена Игоревна К3343

Руководитель: Говоров А. И.

Подписи членов комиссии:

\_\_\_\_\_  
**Ф.И.О.**

(подпись)

\_\_\_\_\_  
**Ф.И.О.**

(подпись)

\_\_\_\_\_  
**Ф.И.О.**

(подпись)

Дата \_\_\_\_

Санкт-Петербург  
2020

# СОДЕРЖАНИЕ

СОДЕРЖАНИЕ .....	2
ВВЕДЕНИЕ .....	3
1. СТРУКТУРА ПРОЕКТА.....	4
1.1 Сведения об используемых средствах разработки .....	4
1.2 Описание модели данных.....	5
2. РЕАЛИЗАЦИЯ WEB-СЕРВИСА.....	6
2.1. Проектирование приложения.....	6
2.2. Серверная часть приложения .....	7
2.3. Клиентская часть приложения. ....	13
ЗАКЛЮЧЕНИЕ .....	19
СПИСОК ИСТОЧНИКОВ .....	20

## ВВЕДЕНИЕ

Цель выполнения курсовой работы заключалась в овладении практическими навыками и умениями реализации web-сервисов средствами Django REST framework, Vue.js.

Задачи курсовой работы:

1. Проанализировать предметную область.
2. Создать модель данных.
3. Реализовать серверную часть средствами Django REST framework.
4. Реализовать клиентскую часть средствами Vue.js.

Была создана программная система, предназначенная для администратора лечебной клиники. Прием пациентов ведут несколько врачей различных специализаций. На каждого пациента клиники заводится медицинская карта, в которой отражается вся информация по личным данным больного и истории его заболеваний. При очередном посещении врача в карте отражается дата и время приема, диагноз, текущее состояние больного, рекомендации по лечению. Так как прием ведется только на коммерческой основе, после очередного посещения пациент должен оплатить медицинские услуги. Расчет стоимости посещения определяется врачом согласно прейскуранту по клинике.

Для ведения внутренней отчетности необходима следующая информация о врачах (фамилия, имя, отчество, специальность, образование, пол, дата рождения и дата начала и окончания работы в клинике, данные по трудовому договору). Для каждого врача составляется график работы с указанием рабочих и выходных дней.

Back-end данного сервиса был реализован с помощью Django Rest Framework, а Front-end – с помощью Vue.js. Также в качестве базы данных была использована PostgreSQL.

# **1. СТРУКТУРА ПРОЕКТА.**

## **1.1 Сведения об используемых средствах разработки**

Для реализации программной системы был выбран Django Rest Framework. Это удобный инструмент для работы с REST, который предоставляет готовую архитектуру для разработки как простых RESTful API, так и более сложных конструкций[4]. Его ключевая особенность – это четкое разделение на сериализаторы, которые описывают соответствие между моделью и ее форматом представления (JSON, XML и др.), и на отдельный набор универсальных представлений на основе классов (Class-Based-Views), которые могут быть по необходимости расширены[4]. С помощью этого инструмента было реализовано отображение данных, выборка нужной информации и добавление новой, а также запросы.

Vue.js использовался для создания пользовательских интерфейсов. Это прогрессивный фреймворк, который пригоден для постепенного внедрения. Его ядро в первую очередь решает задачи уровня представления (view), что упрощает интеграцию с другими библиотеками и существующими проектами. Представления web-сервиса отображены в папке Components и полностью удовлетворяют запрашиваемому функционалу.

В качестве базы данных была использована PostgreSQL. Это свободная объектно-реляционная система управления базами данных, базирующаяся на языке SQL и поддерживающая многочисленные возможности. СУБД отличается высокой надёжностью и хорошей производительностью.

## 1.2 Описание модели данных.

Для БД разрабатываемого web-сервиса были выбраны следующие сущности:

- Пациент
- Врач
- Прейскурант
- Прием
- Оплата приема
- Данные о приеме
- График работы
- Кабинет
- Медицинская карта

Реализация вышеописанной базы данных представлена на Рисунке 1.

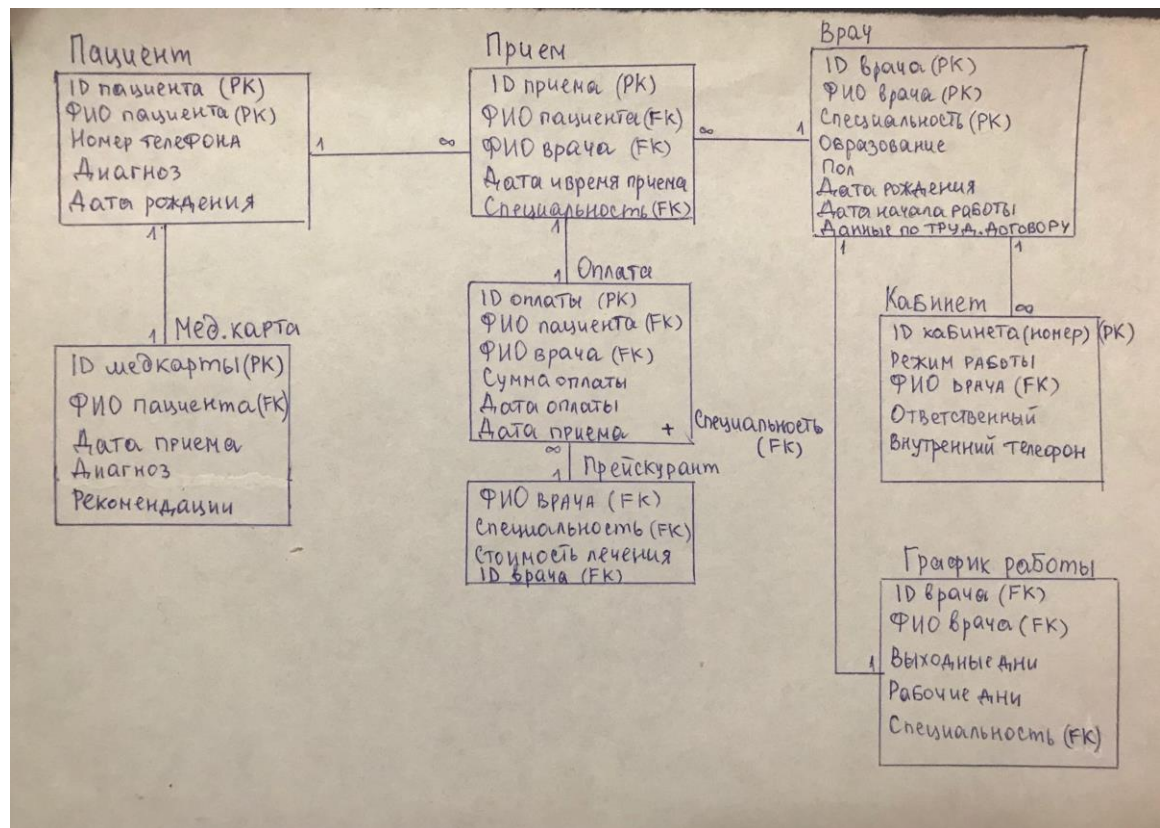


Рисунок 1. Модель созданной БД.

## 2. РЕАЛИЗАЦИЯ WEB-СЕРВИСА

### 2.1. Проектирование приложения

Проект содержит следующие основные интерфейсы: Главная страница, Моя медкарта, Запись на прием, Ведение приема и Выход. Для удобства слева находится боковая панель, с помощью которой можно переключаться между этими интерфейсами.

Описание функционала полученного WEB-сервиса:

1. Фильтрация пациентов по врачам с датами и стоимостью приёмов.
2. Вывод определённой информации по пациентам, которые посещали заданного врача с использованием дополнительных параметров поиска (например, дата рождения).
3. Получение списка врачей, работающих в заданный день.
4. Статистика по количеству приёмов в определённую дату.
5. Суммарная стоимость услуг клиники по дням и врачам.
6. Список пациентов, уже оплативших лечение.

В проекте используется клиент-серверная архитектура. Основной принцип технологии "клиент-сервер" заключается в разделении функций приложения на три группы[7]:

- ввод и отображение данных (взаимодействие с пользователем);
- прикладные функции, характерные для данной предметной области;
- функции управления ресурсами (файловой системой, базой данных и т.д.)

Схема взаимодействия компонентов представлена на рисунке ниже.

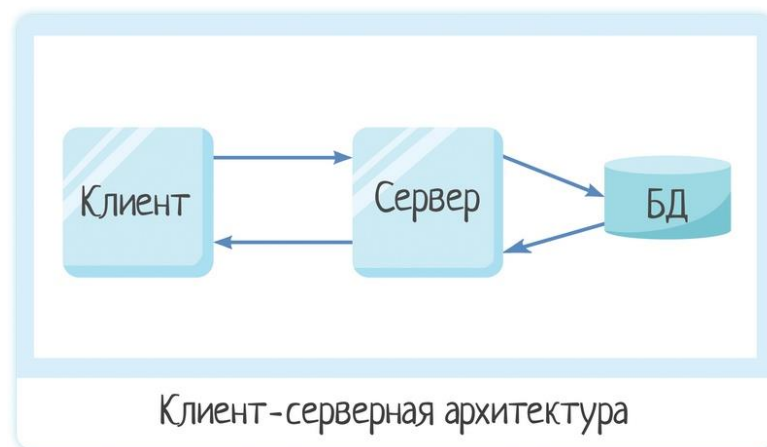


Рисунок 2.

## 2.2. Серверная часть приложения

Серверная часть приложения реализуется с помощью фреймворка Django REST. Создается файл, содержащий описание таблиц базы данных, представленное в виде класса Python – модель. Были использованы следующие модули:

- django
- django rest framework
- psycopg2
- django cors headers

Были добавлены следующие модели:








HOSPITAL		
Appointments	+ Add	 Change
Cabinets	+ Add	 Change
Calendars	+ Add	 Change
Doctors	+ Add	 Change
Patient cards	+ Add	 Change
Patients	+ Add	 Change
Payments	+ Add	 Change
Prices	+ Add	 Change

Рисунок 3. Добавление моделей.

Далее было положено начало созданию интерфейсов.

На рисунках 4-7 представлен процесс заполнения базы данных, на рисунках 8-9 – формы для удобного добавления данных, а на рисунке 10 – форма для изменения и обновления данных.

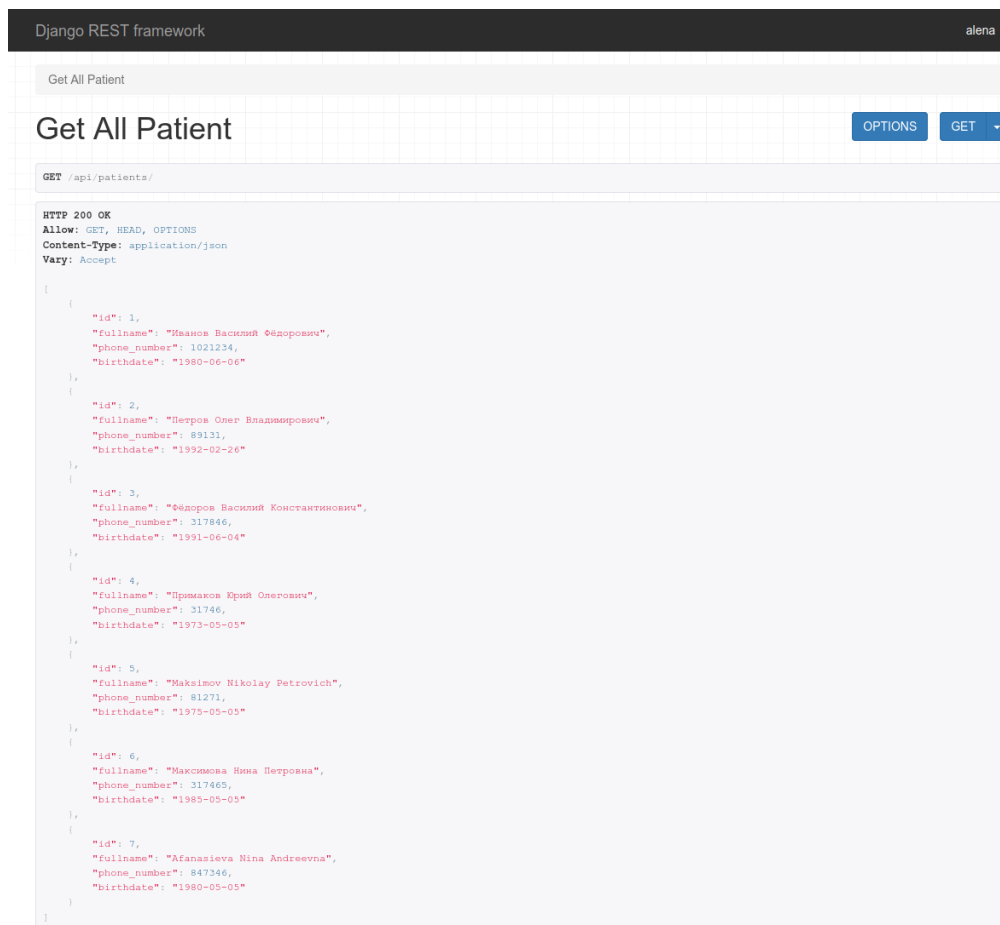


Рисунок 4.

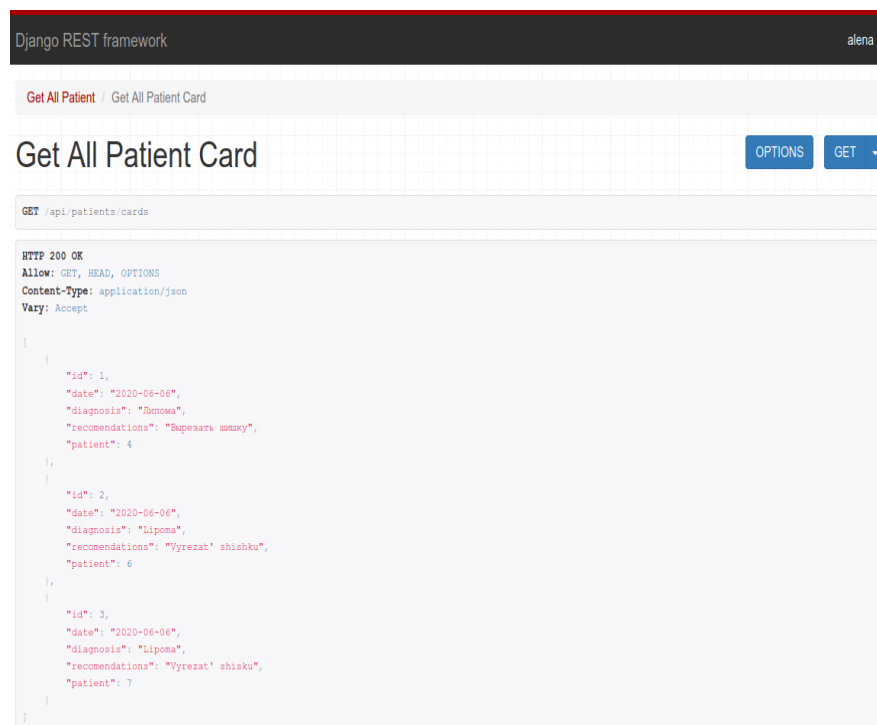


Рисунок 5.



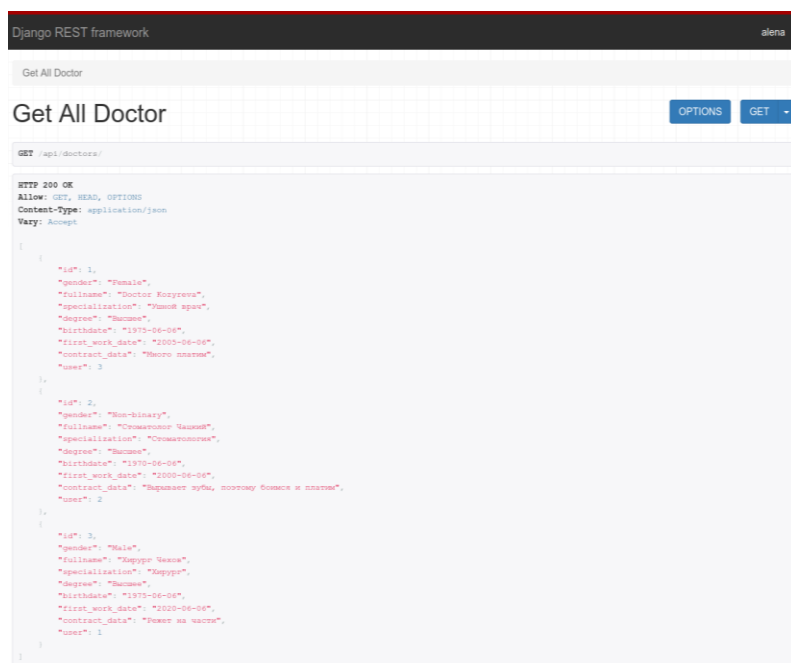


Рисунок 6.

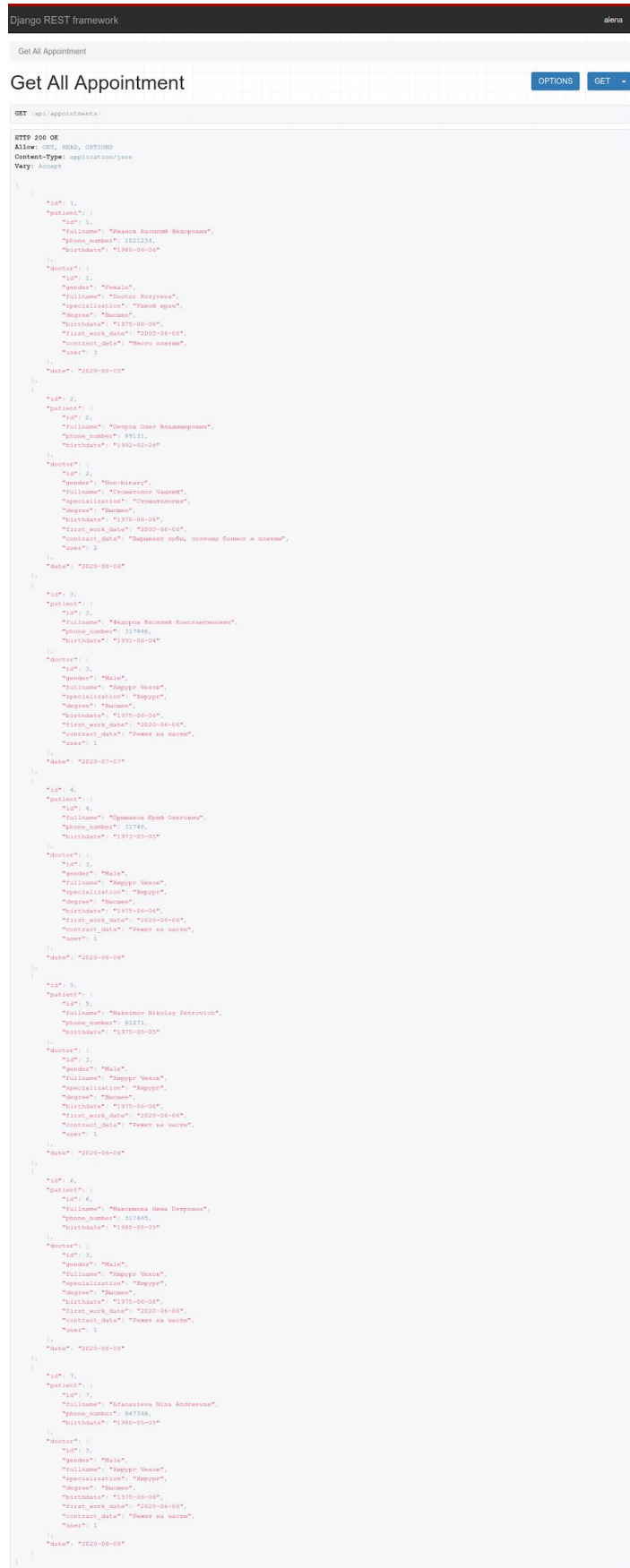


Рисунок 7.

Django REST frameworkalena

Get All Appointment / Create Appointment

Create AppointmentOPTIONS

GET /api/appointments/add/

**HTTP 405 Method Not Allowed**  
Allow: POST, OPTIONS  
Content-Type: application/json  
Vary: Accept  

```
{
  "detail": "Method \"GET\" not allowed."
}
```

Raw data HTML form

Date

Doctor

Patient

POST

Рисунок 8.

Django REST frameworkalena

Get All Patient / Create Patient

Create PatientOPTIONS

GET /api/patients/add/

**HTTP 405 Method Not Allowed**  
Allow: POST, OPTIONS  
Content-Type: application/json  
Vary: Accept  

```
{
  "detail": "Method \"GET\" not allowed."
}
```

Raw data HTML form

Fullname

Phone number

Birthdate

POST

Рисунок 9.

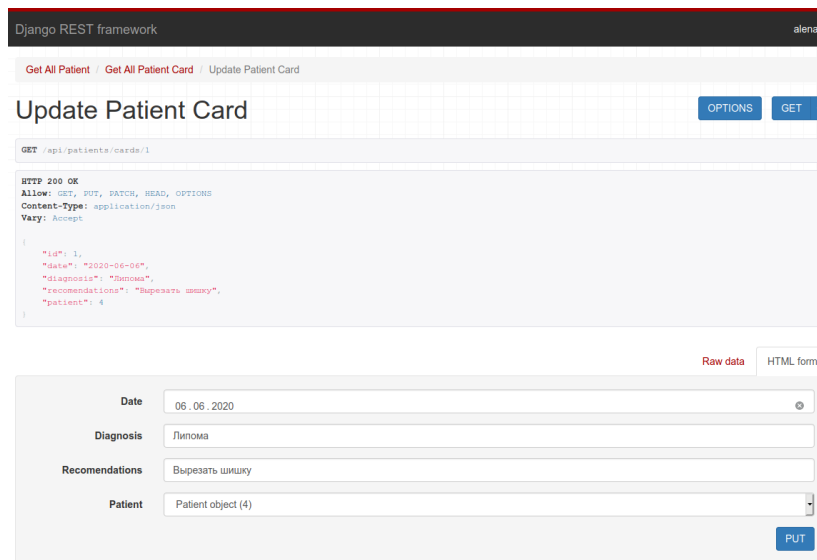


Рисунок 10.

Так как используется Django Rest Framework, нужно определить сериализаторы. Сериализаторы позволяют преобразовывать сложные данные, такие как наборы запросов и экземпляры модели, в собственные типы данных Python, которые затем можно легко преобразовать в JSON, XML или другие типы содержимого. Сериализаторы также обеспечивают обратное сложные типы после предварительной проверки входящих данных.

В каждом сериализаторе нужно указать поля, которые будут передавать в представление, используя уже составленные модели. После сериализации нужно создать представление для каждой модели в файле views.py.

В этом же файле описываются методы GET и POST, с помощью которых клиент и сервер могут обмениваться данными.

Для метода POST используется проверка введенных данных во избежание конфликтов в базе данных.

### 2.3. Клиентская часть приложения.

С преподавателем были согласованы следующие интерфейсы:

1. Главная страница и авторизация.
2. Интерфейс записи на прием.
3. Интерфейс ведения приема.
4. Личный кабинет пациента
5. Медкарта пациента с возможностью посмотреть историю приемов и оплатить прием.

Интерфейсы реализуются с помощью технологий Vue.js и Bootstrap. Файл Bootstrap подключается в файле index.html проекта, интерфейсы описываются как template в файлах-компонентах формата vue.

Для начала стоит упомянуть об имеющихся эндпойнтах и их предназначении. Они представлены на рисунке ниже.

```
Using the URLconf defined in django_hospital.urls, Django tried these URL patterns, in this order:  
1. admin/  
2. api/ auth/  
3. api/ auth/token [name='token']  
4. api/ patients/  
5. api/ patients/cards  
6. api/ doctors/  
7. api/ appointments/  
8. api/ payments/  
9. api/ prices/  
10. api/ appointments/<int:pk>  
11. api/ payments/<int:pk>  
12. api/ appointments/add/  
13. api/ payments/add/  
14. api/ doctors/add/  
15. api/ patients/add/  
16. api/ patients/cards/add/  
17. api/ patients/cards/<int:pk>
```

Рисунок 11. Список эндпойнтов.

Эндпойнты, заканчивающиеся на «s», выводят все данные из соответствующей таблицы.

Заканчивающиеся на «int:pk» выводят данные о конкретной записи (как правило, по первичному ключу или по id).

Те эндпойнты, которые заканчиваются на «add», отвечают за добавление новых записей в какую-либо таблицу.

Сортировка и выборка необходимых данных происходит на back-енде.

Далее рассмотрим интерфейсы более подробно.

#### А) Главная страница

На данном интерфейсе мы можем посмотреть приемы и всю информацию о них, а также найти прием по заданным параметрам, таким как:

- Врач
- Дата
- Стоимость приема
- Наличие оплаты
- Пациент
- Дата рождения пациента

Также здесь есть возможность вывести суммарную стоимость приемов и, после получения необходимой информации, нажать кнопку «Сбросить». Данный интерфейс представлен на Рисунке 12.

The screenshot shows a web application interface for a hospital. At the top is a green header with a menu icon and the text 'Hospital'. Below the header, the title 'Клиника "Hospital"' is displayed. The main section is titled 'Найти пациента' (Find patient). It contains several search filters: 'Лечащий врач' (Attending doctor) with a dropdown arrow, 'Дата приёма' (Appointment date) with a date input field showing 'ДД.ММ.ГГГГ', 'Стоимость приёма' (Appointment cost) with a text input field, 'Дата рождения пациента' (Patient's date of birth) with a date input field showing 'ДД.ММ.ГГГГ', and a checkbox 'Была ли оплата?' (Was there a payment?). There are also two buttons: 'ОТФИЛЬТРОВАТЬ' (Filter) and 'СБРОСИТЬ' (Reset). Below the filters, it says 'Найдено 4 приёмов' (4 appointments found). A list of appointments is shown in a table with green rows. Each row contains the patient's name, their role and last name, and the cost of the appointment.

Пациент	Врач	Стоимость
Иванов Василий Фёдорович	Doctor Kazuyeva	1500
Примаков Юрий Олегович	Хирург Чехов	2000
Максимова Нина Петровна	Хирург Чехов	5000
Afanasieva Nina Andreevna	Хирург Чехов	1500

Рисунок 12. Главная страница

#### Б) Авторизация

Вполне логичным дальнейшим действием может быть запись на прием. Но невозможно записаться на прием, не авторизовавшись. Поэтому при нажатии на кнопку

«Запись на прием» открывается страница входа (Рисунок 13).

В случае, если пользователь еще не зарегистрирован, можно перейти на страницу регистрации (Рисунок 14). Здесь необходимо указать логин, пароль, ФИО, номер телефона и дату рождения. Пароль секретный – при его вводе символы не отображаются на экране. После успешного прохождения регистрации снова откроется страница входа – здесь нужно будет ввести логин и пароль.

Рисунок 13. Интерфейс входа.

Рисунок 14. Интерфейс авторизации.

### В) Интерфейс «Медицинская карта».

После входа в систему сразу открывается медицинская карта пользователя. Здесь можно увидеть информацию о приемах – дату приемов, а также диагноз и рекомендации (при их наличии). Данный интерфейс представлен на Рисунке 15.

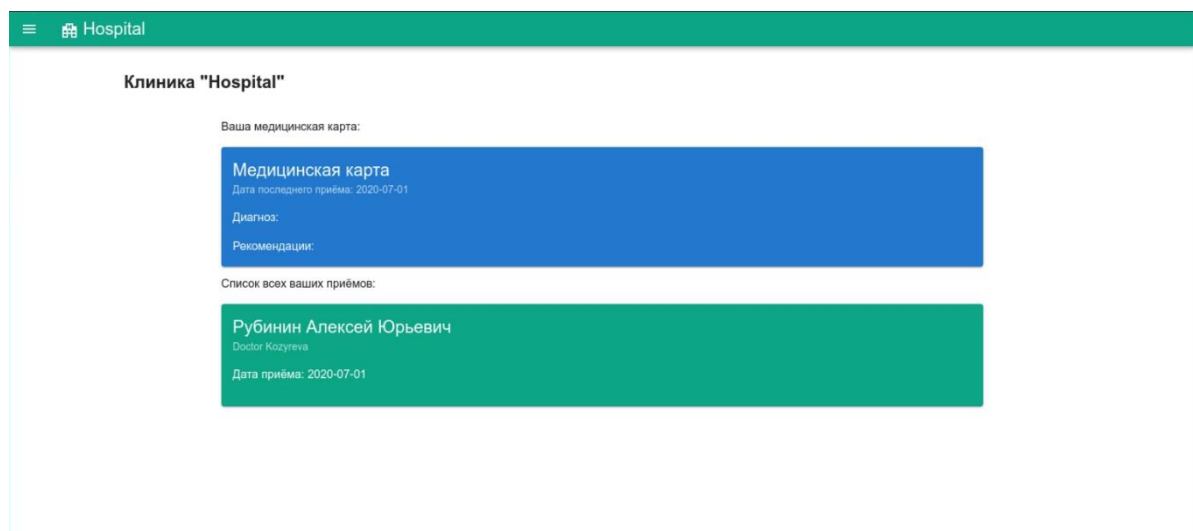


Рисунок 15. Интерфейс «Медкарта».

Г) Интерфейс «Запись на прием».

Данный интерфейс представлен на рисунке 16. В нем необходимо указать желаемую дату приема и врача. После этого нужно нажать на кнопку «Отправить» и появится окно, сообщающее о том, что с пользователем свяжутся через 5 минут (Рисунок 17).

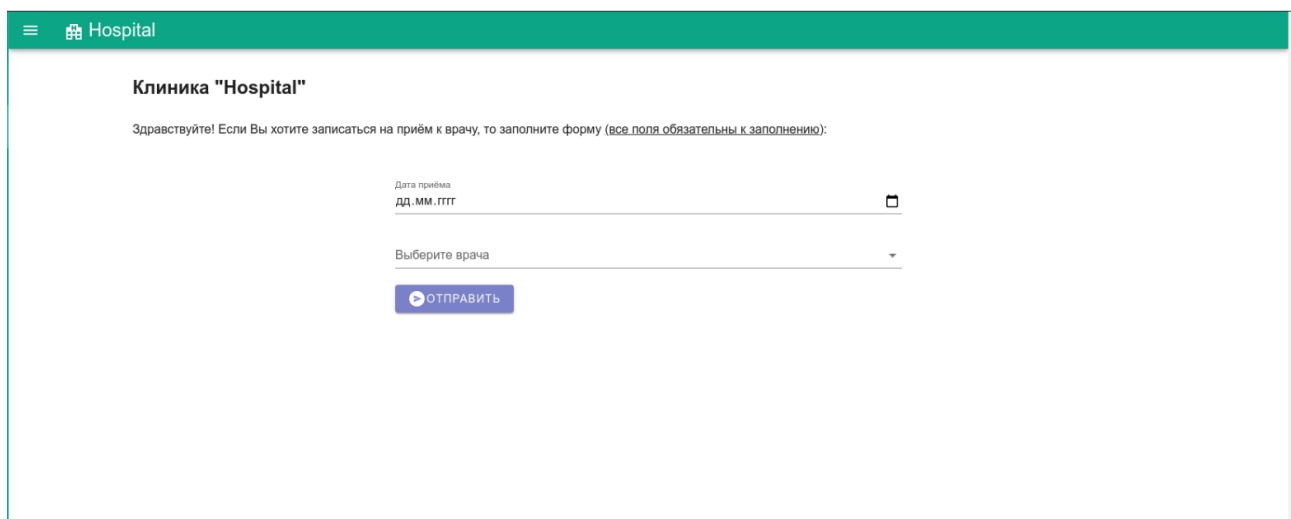


Рисунок 16.

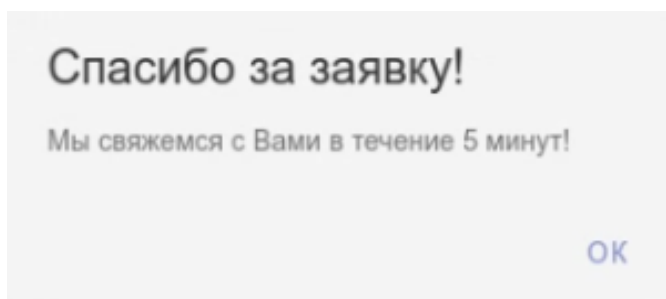


Рисунок 17.



#### Д) Интерфейс «Ведение приема».

Данный интерфейс предназначен для врачей. Здесь есть возможность указать более подробную информацию о приеме, а именно: стоимость приема, диагноз пациента и выданные рекомендации. После этого необходимо нажать кнопку «Отправить», чтобы данные зафиксировались в базе, после чего появится уведомление об этом (Рисунок 19). Интерфейс представлен на Рисунке 18.

The screenshot shows a web interface for a clinic named "Hospital". The header is teal with a menu icon and the text "Hospital". Below the header, the text "Клиника "Hospital"" and "Приём номер 4" is displayed. A message says: "Здравствуйте! Вам следует зафиксировать информацию о приёме в форме (все поля обязательны к заполнению):". Below this is a form with three input fields labeled "Стоимость", "Диагноз", and "Рекомендации". At the bottom of the form is a blue button with a white arrow and the text "ОТПРАВИТЬ".

Рисунок 18. Ведение приема.

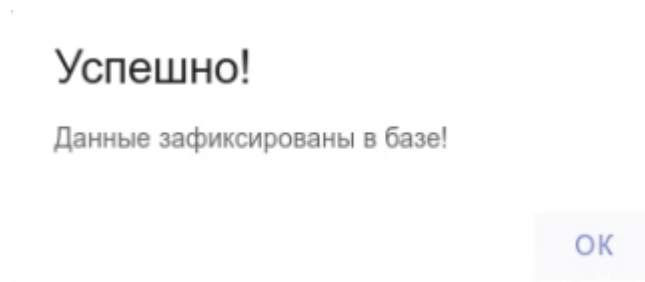


Рисунок 19.

#### Е) Боковое меню

Для удобства навигации между интерфейсами было создано боковое меню. Здесь есть возможность переключаться с одной страницы на другую. Всего есть 5 пунктов: Главная страница, Медкарта, Запись на прием, Ведение приема и Выход из системы. Данная панель представлена на Рисунке 20.

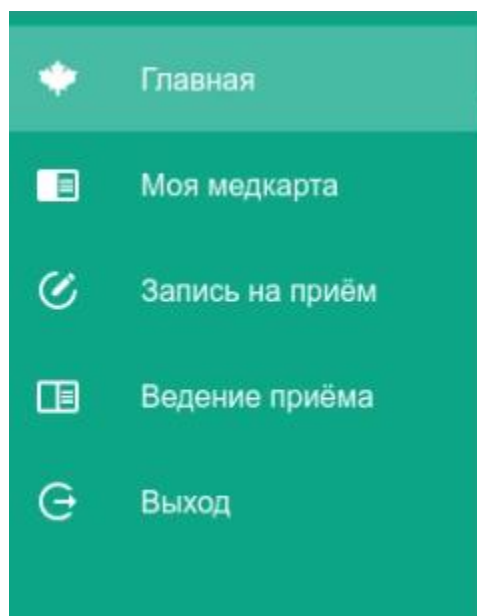


Рисунок 20. Боковая панель.

## **ЗАКЛЮЧЕНИЕ**

Цель курсовой работы заключалась в создании программной системы, предназначенной для администратора лечебной клиники.

В соответствии с индивидуальным заданием было реализовано веб-приложение с использованием технологий Django REST Framework, Vue.js и PostgreSQL. В приложении реализован набор заданных интерфейсов. Также в ходе выполнения работы был получен опыт разработки front-end и back-end частей приложения, практические навыки веб-разработки. В результате выполнения работы удалось выполнить все поставленные задачи, а, следовательно, цель достигнута.

## СПИСОК ИСТОЧНИКОВ

- 1.Официальный сайт Django Rest Framework. Документация Django Rest Framework [Электронный ресурс]. URL: <https://www.django-rest-framework.org> (дата обращения: 30.06.2020).
- 2.Документация Django на русском языке [Электронный ресурс] URL: <https://djbook.ru/rel1.9/> (Дата обращения: 30.06.2020).
- 3.Vue.js. Введение [Электронный ресурс] URL: <https://ru.vuejs.org/v2/guide/index.html> (Дата обращения 01.07.2020)
- 4.Хабр. Web API с помощью Django REST framework. [Электронный ресурс] URL: <https://habr.com/ru/post/160117/>.(Дата обращения 30.06.2020)
- 5.Asyncee. Продвинутые запросы в Django: сортировка по дате [Электронный ресурс]. URL: <https://asyncee.github.io/posts/advanced-django-querying-sorting-events-by-date/> (дата обращения: 30.06.2020).
- 6.Хабр. Клиент-серверная архитектура в картинках [Электронный ресурс]. URL: <https://habr.com/ru/post/495698/> (Дата обращения: 06.07.2020)
- 7.Архитектура "Клиент-сервер" [Электронный ресурс]. URL: [http://www.mstu.edu.ru/study/materials/zelenkov/ch\\_7\\_1.html](http://www.mstu.edu.ru/study/materials/zelenkov/ch_7_1.html) (Дата обращения: 06.07.2020).