|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ | | | | |
| федеральное государственное бюджетное образовательное учреждение | | | | |
| «Балтийский государственный технический университет «ВОЕНМЕХ» им. Д.Ф. Устинова» | | | | |
| (БГТУ «ВОЕНМЕХ» им. Д.Ф. Устинова») | | | | |
| БГТУ.СМК-Ф-4.2-К5-01 | | | | |
| Факультет | | | О |  | Естественнонаучный |
|  |  |  | шифр |  | наименование |
| Кафедра | | | О7 |  | Информационные системы и программная инженерия |
|  |  |  | шифр |  | наименование |
| Дисциплина | |  | Базы данных | | |

КУРСОВОЙ ПРОЕКТ

на тему

|  |
| --- |
| Разработка ИПк для Поликлиники |
|  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент группы | | | | | | |  | И508Б |
|  |  | Кабиров К. Р. | | | | | | |
|  |  | Фамилия И.О. | | | | |  |  |
|  |  | **РУКОВОДИТЕЛЬ** | | | | | | |
| Заборовский И.С | | | | | | |  |  |
| Фамилия И.О. | | | | |  |  | Подпись | |
| Оценка | | | | | | |  |  |
| « » | | |  |  |  |  |  | 2023 г. |

**РЕФЕРАТ**

Отчет 34 с., 27 рис., 4 табл., 5 источн., 1 прил.

ДАТАЛОГИЧЕСКАЯ МОДЕЛЬ, ИНФОЛОГИЧЕСКАЯ МОДЕЛЬ, БАЗА ДАННЫХ, СУЩНОСТЬ, СВЯЗЬ

Объект разработки – база данных и клиентский интерфейс для поликлиники.

Целью курсового проекта является разработка базы данных и клиентского приложения для поликлиники.

В процессе работы проводились исследования предметной области, разрабатывались инфологические и даталогические схемы.

В результате разработки была получена заполненная база данных, а также клиентское приложения для взаимодействия с базой данных путем вывода результатов запроса в графическом виде.

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc136863991)

[1 Обзор предметной области 6](#_Toc136863992)

[2 Инфологическое моделирование предметной области 8](#_Toc136863993)

[2.1 Спецификация сущностей 9](#_Toc136863994)

[2.2 Спецификация связей 9](#_Toc136863995)

[2.3 Спецификация атрибутов 10](#_Toc136863996)

[2.4 Инфологическая схема 13](#_Toc136863997)

[3 Даталогическое проектирование базы данных 14](#_Toc136863998)

[3.1 Даталогическая схема 14](#_Toc136863999)

[3.2 Спецификация ключей 16](#_Toc136864000)

[4 Выбор средств разработки 17](#_Toc136864001)

[5 Проектирование БД и описание запросов 18](#_Toc136864002)

[6 Создание клиентской части ИС 22](#_Toc136864003)

[ЗАКЛЮЧЕНИЕ 32](#_Toc136864004)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 33](#_Toc136864005)

[ПРИЛОЖЕНИЕ А 34](#_Toc136864006)

# ВВЕДЕНИЕ

Термин «база данных» не имеет конкретного определения, однако его можно сформулировать следующим образом: база данных — совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняется в соответствии с правилами средств моделирования данных. [1] Одной из главных задач баз данных является обеспечение доступа и управления данными, что позволяет пользователям эффективно работать с большими объемами информации.

Базы данных можно разделить на реляционные и нереляционные.

Реляционная база данных – это структура данных, в которой данные хранятся в виде таблиц, состоящих из строк и столбцов. Реляционные базы данных используют язык SQL (Structured Query Language) для управления данными.

Нереляционная база данных (NoSQL)- это база данных, которая не использует традиционную структуру таблиц и соответствующих связей между ними. Вместо этого использует другие типы хранилищ данных, такие как документы, графы, пары ключ/значение или колоночные хранилища.

Целью курсового проекта является разработка ИПК для поликлиники. Для достижения поставленной цели необходимо решить следующие задачи:

* изучить и описать предметную область, в рамках которой выполняется курсовой проект;
* разработать инфологическую модель данных анализируемой предметной области;
* преобразовать инфологическую модель в даталогическую реляционную модель данных;
* обеспечить перевод даталогической модели в физическую среду конкретной СУБД;
* наполнить разработанную БД данными;
* составить запросы к БД, согласно варианту задания.
* реализовать пользовательский интерфейс;
* протестировать работу программы.

Отчёт состоит из введения, 6 разделов и заключения.

Ниже представлено краткое описание содержания каждого из разделов.

Обзор предметной области – описание составляющих железной дороги, которые используются в инфологическом моделировании.

Инфологическое моделирование предметной области – определение сущностей и связей между ними.

Даталогическое проектирование базы данных – на основе полученных инфологической модели строится даталогическая модель БД.

Выбор средств разработки – для создания ИПК использовался язык программирования Python фреймворк Django и СУБД MySQL.

Проектирование БД и описание запросов – проектирование БД согласно даталогической модели, описание запросов, использующихся в клиентской части.

Создание клиентской части ИС – создание приложения с графическим интерфейсом и демонстрация его работы.

# 1 Обзор предметной области

Система управления поликлиникой была разработана для управления различными бизнес-процессами, связанными с записью пациентов на прием к врачам, выписывания счетов за прием. Для этих целей необходима база данных, содержащая подробную информацию о каждом элементе поликлиники:

База данных хранит следующую информацию:

* врач – ФИО, специальность, номер телефона, адрес электронной почты, адрес проживания.
* страховая компания – название, номер телефона, адрес
* пациент – ФИО, номер телефона, адрес проживания, страховая компания, к которой принадлежит пациент;
* прием у врача – врач, который проводил прием, пациент, который записался на него, дата и время начала приема;
* счет – прием за который выписан счет, застрахован ли пациент, дата отправления счета, сумма;
* платеж – пациент, страховая компания, счет, сумма.

Бизнес-процессы, которые могут быть автоматизированы этой системой, включают в себя:

* управление записью на прием;
* управление расписанием врачей;
* управление страховыми данными;
* выставление счетов и управление платежами;
* аналитика и отчетность
* уведомления и напоминания.

Система автоматизации, основанная на этих данных о поликлинике, может быть полезна предоставлена следующим группам пользователей:

Администрация поликлиники: Система может быть предоставлена администраторам и руководителям поликлиники для управления операциями, мониторинга процессов, принятия решений и получения аналитической информации.

Врачи и медицинский персонал: Система может быть доступна для врачей и медицинского персонала для управления расписанием, записью на прием, ведением медицинских записей, выписывания рецептов и назначений, а также получения уведомлений и напоминаний.

Пациенты: Пациенты могут получать доступ к системе для онлайн-записи на прием, просмотра своих медицинских записей, получения уведомлений о предстоящих приемах, оплате счетов и обновлении информации о страховке.

Страховые компании: Система может предоставить доступ страховым компаниям для получения информации о пациентах, оплате счетов, управлениями платежами и обновлении данных о страховке.

Рассмотрим вариант работы с системой для администрации поликлиники:

Администратор может использовать систему для установки расписания работы врачей. Он выбирает врача, дату и время начала и окончания приема, а затем добавляет эту информацию в систему. Расписание врача обновляется, и пациенты могут видеть доступные слоты для записи на прием.

Кроме того, система предоставляет администрации возможность генерировать различные отчеты и проводить анализ данных. Например, администратор может создать отчет о приемах врачей за определенный период, анализировать статистику посещаемости или оценивать эффективность работы отдельных врачей на основе количества приемов.

# 2 Инфологическое моделирование предметной области

Поликлиника является важным звеном в здравоохранении, предоставляющим медицинские услуги населению. Для эффективной работы поликлиники необходимо иметь хорошо организованную систему управления и обработки данных. Инфологическое моделирование предметной области поликлиники помогает нам лучше понять все аспекты и процессы, связанные с оказанием медицинской помощи пациентам.

Инфологическая модель поликлиники включает следующие сущности и их атрибуты:

* врач: содержит информацию о враче, включая ФИО, специальность, номер телефона, адрес электронной почты и адрес проживания.
* страховая компания: хранит информацию о страховой компании, включая название, номер телефона и адрес.
* пациент: хранит информацию о пациенте, включая ФИО, номер телефона, адрес проживания и страховую компанию, к которой он принадлежит.
* прием у врача: связывает врача и пациента, а также содержит информацию о дате и времени начала приема.
* счет: отражает прием, за который выписан счет, включая информацию о застрахованности пациента, дате отправления счета и сумме.
* платеж: связывает пациента, страховую компанию и счет, а также содержит информацию о сумме платежа.

Инфологическая модель поликлиники помогает управлять информацией о пациентах, расписании врачей, оплате услуг и взаимодействии со страховыми компаниями. Она облегчает процессы записи на прием, выписывания счетов, учета платежей и анализа данных. Благодаря такой модели администрация поликлиники может оптимизировать работу, улучшить обслуживание пациентов и принимать информированные решения, основанные на анализе собранных данных.

## 2.1 Спецификация сущностей

В инфологической модели данных определяются сущности и связи между ними. [2] Сущности представляют отдельные объекты в предметной области, такие как врачи, пациенты, приемы, страховые компании, счета, платежи.

В таблице 1 представлено описание сущностей поликлиники

Таблица 1 – Описание сущностей поликлиники

|  |  |  |
| --- | --- | --- |
| Сущность | Уникальный идентификатор | Описательные атрибуты |
| Специальность | Идентификатор специальности | Название специальности, описание специальности |
| Расписание | Идентификатор расписания | Врач, день недели, начало приема, конец приема |
| Врач | Идентификатор доктора | ФИО врача, специальность, номер телефона, адрес электронной почты, адрес проживания |
| Талон | Идентификатор талона | Врач, дата, время, пациент |
| Пациент | Идентификатор пациента | ФИО пациента, номер телефона, адрес проживания, страховая компания, к которой принадлежит пациент |
| Страховая компания | Идентификатор страховой компании | Название страховой компании, номер телефона, адрес |
| Прием у врача | Идентификатор приема | Врач, проводящий прием, пациент, записавшийся на прием, дата и время начала приема |
| Счет | Идентификатор счета | Прием, за который выписан счет, застрахован ли пациент, дата отправления счета, сумма |
| Платеж | Идентификатор платежа | Пациент, страховая компания, счет, сумма |

## 2.2 Спецификация связей

Связи определяют отношения между сущностями. Это могут быть следующие отношения:

* один-к-одному (1:1) – экземпляр одной сущности связан только с одним экземпляром другой сущности;
* один-ко-многим (1:М) – один экземпляр сущности, расположенный слева по связи, может быть связан с несколькими экземплярами сущности, расположенными справа по связи;
* многие-ко-многим (М:М) – что один экземпляр первой сущности может быть связан с несколькими экземплярами второй сущности, и наоборот, один экземпляр второй сущности может быть связан с несколькими экземплярами первой сущности. Как правило, связь многие-ко-многим выполняется с использованием промежуточной таблицы.

В таблице 2 представлено описание связей поликлиники.

Таблица 2 – Описание связей поликлиники

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер | Связь | Тип | От сущности | К сущности | По атрибуту |
| 1 | Имеет | 1:1 | Врач | Специальность | Ид. Специальности |
| 2 | Врач | 1:m | Врач | Расписание | Ид. Врача |
| 3 | Ведет | 1:М | Врач | Прием у врача | Ид. Врача |
| 4 | Записывается | 1:М | Пациент | Прием у врача | Ид. Пациента |
| 5 | Застрахован | 1:М | Страховая компания | Пациент | Ид. Страховой компании |
| 6 | Генерирует | 1:1 | Прием у врача | Счет | Ид. Приема |
| 7 | Талон | 1:1 | Пациент | Принадлежит | Ид. Пациента |
| 5 | Погашает | 1:М | Платеж | Счет | Ид. Платежа |
| 6 | Производит | 1:М | Пациент | Платеж | Ид. Пациента |
| 7 | Производит | 1:М | Страховая компания | Платеж | Ид. Страховой компании |

## 2.3 Спецификация атрибутов

Спецификация атрибутов в базе данных – это процесс определения характеристик каждого атрибута в таблице базы данных. Атрибуты могут быть числовыми, строковыми, датами или другими типами данных, и каждый атрибут должен иметь свою уникальную спецификацию.

Спецификация атрибутов включает в себя описание каждого атрибута, включая его название, тип данных, длину поля и ограничения на значения. Например, если атрибут содержит целочисленные значения, то он может иметь ограничение на минимальное или максимальное значение. Если атрибут содержит текстовые значения, то он может иметь ограничение на максимальную длину строки. В спецификации атрибутов также может быть указано, является ли атрибут обязательным для заполнения или нет, и может ли он содержать повторяющиеся значения или нет. Некоторые атрибуты могут также иметь ссылки на другие таблицы базы данных или на внешние файлы.

В таблице 3 представлена спецификация атрибутов

Таблица 3 – Спецификация атрибутов

|  |  |  |  |
| --- | --- | --- | --- |
| Сущность | Атрибут | Тип | Ограничения |
| Специальность | Ид специальности | Число | Первичный ключ, ненулевое, до 11 знаков |
| Специальность | название | Строка | Ненулевое, до 32 символов |
| Специальность | Название | Строка | Ненулевое, до 128 символов |
| Врач | Ид врача | Число | Первичный ключ, ненулевое, до 11 знаков |
| Врач | Имя | Строка | Ненулевое, до 32 символов |
| Врач | Фамилия | Строка | Ненулевое, до 32 символов |
| Врач | Отчество | Строка | Ненулевое, до 32 символов |
| Врач | Ид специальности | Число | Вторичный ключ, ненулевое, до 11 знаков |
| Врач | Номер телефона | Строка | Ненулевое, до 64 символов |
| Врач | Адрес эл. Почты | Строка | Ненулевое, до 128 символов |
| Врач | Адрес | Строка | Ненулевое, до 254 символов |
| Расписание | Ид расписания | Число | Первичный ключ, ненулевое, до 11 знаков |
| Расписание | Врач | Ид врача | Число |
| Расписание | День недели | Число | Ненулевое, от 0 до 6 |
| Расписание | Время начала приема | Время | Имеет тип времени, задается по маске чч:мм |
| Расписание | Время конца приема | Время | Имеет тип времени, задается по маске чч:мм |
| Страховая компания | Ид страховой компании | Число | Первичный ключ, ненулевое, до 11 знаков |
| Страховая компания | Название | Строка | Ненулевое, до 128 символов |
| Страховая компания | Номер телефона | Строка | Ненулевое, до 64 символов |
| Страховая компания | Адрес | Строка | Ненулевое, до 254 символов |
| Пациент | Ид пациента | Число | Первичный ключ, ненулевое, до 11 знаков |
| Пациент | Имя | Строка | Ненулевое, до 32 символов |
| Пациент | Фамилия | Строка | Ненулевое, до 32 символов |
| Пациент | Отчество | Строка | Ненулевое, до 32 символов |
| Пациент | Номер телефона | Строка | Ненулевое, до 64 символов |
| Пациент | Адрес эл. Почты | Строка | Ненулевое, до 128 символов |

Продолжение таблицы 3

|  |  |  |  |
| --- | --- | --- | --- |
| Сущность | Атрибут | Тип | Ограничения |
| Пациент | Ид страховой компании | Число | Внешний ключ, до 11 знаков |
| Талон | Ид талона | Число | Первичный ключ, ненулевое, до 11 знаков |
| Талон | Ид врача | Число | Вторичный ключ, ненулевое, до 11 знаков |
| Талон | Ид пациента | Число | Вторичный ключ, ненулевое, до 11 знаков |
| Талон | Время начала приема | Время | Имеет тип времени, задается по маске чч:мм |
| Талон | Время конца приема | Время | Имеет тип времени, задается по маске чч:мм |
| Прием у врача | Ид приема | Число | Первичный ключ, ненулевое, до 11 знаков |
| Прием у врача | Ид врача | Число | Внешний ключ, ненулевое, до 11 знаков |
| Прием у врача | Ид пациента | Число | Внешний ключ, ненулевое, до 11 знаков |
| Прием у врача | Дата и время начала приема | Дата и время | Имеет тип даты и времени, задается по маске ДД.ММ.ГГГГ ЧЧ:ММ |
| Счет | Ид счета | Число | Первичный ключ, ненулевое, до 11 знаков |
| Счет | Ид приема | Число | Внешний ключ, ненулевое, до 11 знаков |
| Счет | Застрахован ли пользователь | Число | Имеет логический тип, принимает значение 1 или 0 |
| Счет | Дата отправки счета | Дата | Имеет тип даты, задается по маске ДД.ММ.ГГГГ |
| Счет | Сумма | Число с плавающей точкой | До 10 знаков до запятой и 2 знака после запятой |
| Платеж | Ид платежа | Число | Первичный ключ, ненулевое, до 11 знаков |
| Платеж | Ид пациента | Строка | Внешний ключ, до 11 знаков |
| Платеж | Ид страховой компании | Число | Внешний ключ, до 11 знаков |
| Платеж | Ид счета | Число | Внешний ключ, ненулевое, до 11 знаков |
| Платеж | Сумма | Число | До 10 знаков до запятой и 2 знака после запятой, ненулевое |
| Платеж | Дата, когда был произведен платеж | Дата | Имеет тип даты, задается по маске ДД.ММ.ГГГГ |

## 2.4 Инфологическая схема

Инфологическая схема поликлиники представлена на рисунке 1.

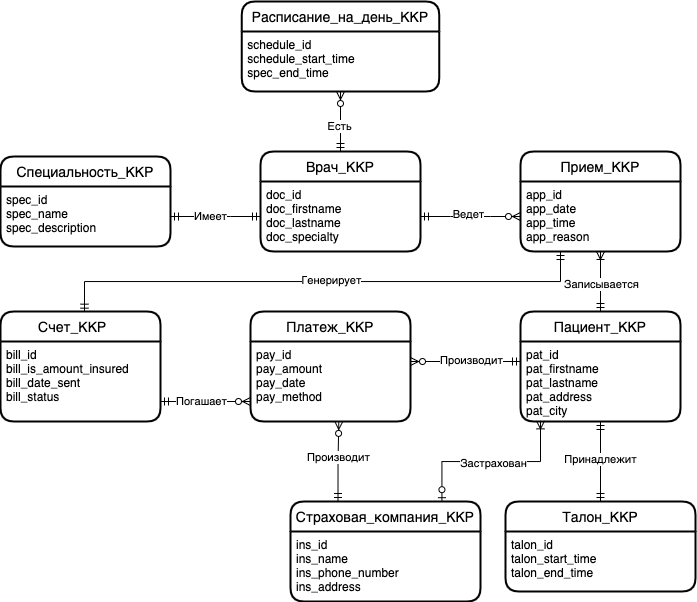


Рисунок 1 – Инфологическая схема железной дороги

Построение данной модели было выполнено с учетом описания сущностей, связей и спецификации атрибутов.

# 3 Даталогическое проектирование базы данных

## 3.1 Даталогическая схема

Даталогическая схема базы данных представлена на рисунке 2.

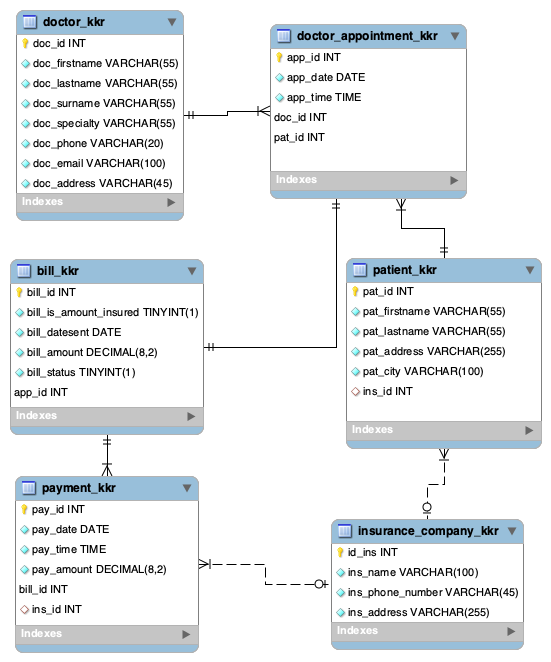


Рисунок 2 – Даталогическая схема базы данных

В таблица 3 представлено соответствие между сущностями и отношениями.

Таблица 3 – Соответствие между сущностями и отношениями

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Ведет | Записывается | Застрахован | Генерирует | Погашает | Производит | Производит | Имеет | Есть |
| Врач | Участвует (1) |  |  |  |  |  |  | Участвует (1) | Участвует (1) |
| Пациент |  | Участвует (1) | Содержит (множ.) |  |  | Участвует (1) |  |  |  |
| Страховая компания |  |  | Участвует (1) | Участвует (1) |  |  | Участвует (1) |  |  |
| Прием у врача | Содержит (множ.) | Содержит (множ.) |  |  |  |  |  |  |  |
| Счет |  |  |  | Участвует (1) | Содержит (множ.) |  |  |  |  |
| Платеж |  |  |  |  | Участвует (1) | Содержит (множ.) | Содержит (множ.) |  |  |
| Специальность |  |  |  |  |  |  |  | Участвует (1) |  |
| Расписание |  |  |  |  |  |  |  |  | Содержит (множ.) |

## 3.2 Спецификация ключей

В таблице 4 представлена спецификация первичных и вторичных ключей.

Таблица 4 – Спецификация первичных и вторичных ключей

|  |  |  |  |
| --- | --- | --- | --- |
| Имя таблицы | Имя поля | Первичный ключ | Вторичный ключ |
| specialty | id | + | – |
| doctor | id | + | \_ |
| doctor | specialty\_id | – | + |
| schedule | id | + | – |
| schedule | doctor\_id | – | + |
| insurance\_company | id | + | – |
| patient | id | + | – |
| patient | insurance\_company\_id | – | + |
| talon | id | + | – |
| talon | patient\_id | – | + |
| talon | doctor\_id | – | + |
| appointment | id | + | – |
| appointment | doctor\_id | – | + |
| appointment | patient\_id | – | + |
| bill | id | + | – |
| bill | appointment\_id | – | + |
| payment | id | + | – |
| payment | insurance\_company\_id | – | + |
| payment | patient\_id | – | + |
| payment | bill\_id | – | + |

# 4 Выбор средств разработки

Были выбраны следующие средства разработки для базы данных и клиентского приложения.

Python – это высокоуровневый, интерпретируемый язык программирования, который широко используется для разработки различных приложений. Он отличается простым и понятным синтаксисом, что делает его доступным даже для начинающих разработчиков. Python обладает обширной стандартной библиотекой, которая включает в себя множество полезных модулей и функций для работы с разными типами данных, файлами, сетью, базами данных и многое другое.

Django – это мощный веб-фреймворк, написанный на языке Python, который позволяет разрабатывать высокопроизводительные веб-приложения. Он предлагает удобные инструменты и функциональность для создания и развертывания веб-сайтов и веб-приложений.

Одним из ключевых преимуществ Django является его простота использования и эффективность. Он предлагает набор готовых компонентов и функций, которые значительно упрощают разработку, таких как система маршрутизации URL, система аутентификации и авторизации, ORM (объектно-реляционное отображение) для работы с базами данных и многое другое. Django также поддерживает шаблонизацию, что позволяет разделять логику приложения и представление данных.

MySQL – это реляционная СУБД с открытым исходным кодом, используемая для хранения данных в различных типах приложений. MySQL имеет широкое распространение и поддерживается большим сообществом разработчиков. Она легко интегрируется с Django, предоставляя надежный и быстрый способ сохранения и извлечения данных [4]

В качестве средств разработки была выбрана среда PyCharm. Она представляет удобную среду разработки для Python и Django, а также поддерживают работу с MySQL.

# 5 Проектирование БД и описание запросов

На рисунках 3-5 представлен процесс создания БД с помощью Django.



Рисунок 3 – Процесс создания базы данных с помощью Django



Рисунок 4 – Процесс создания базы данных с помощью Django

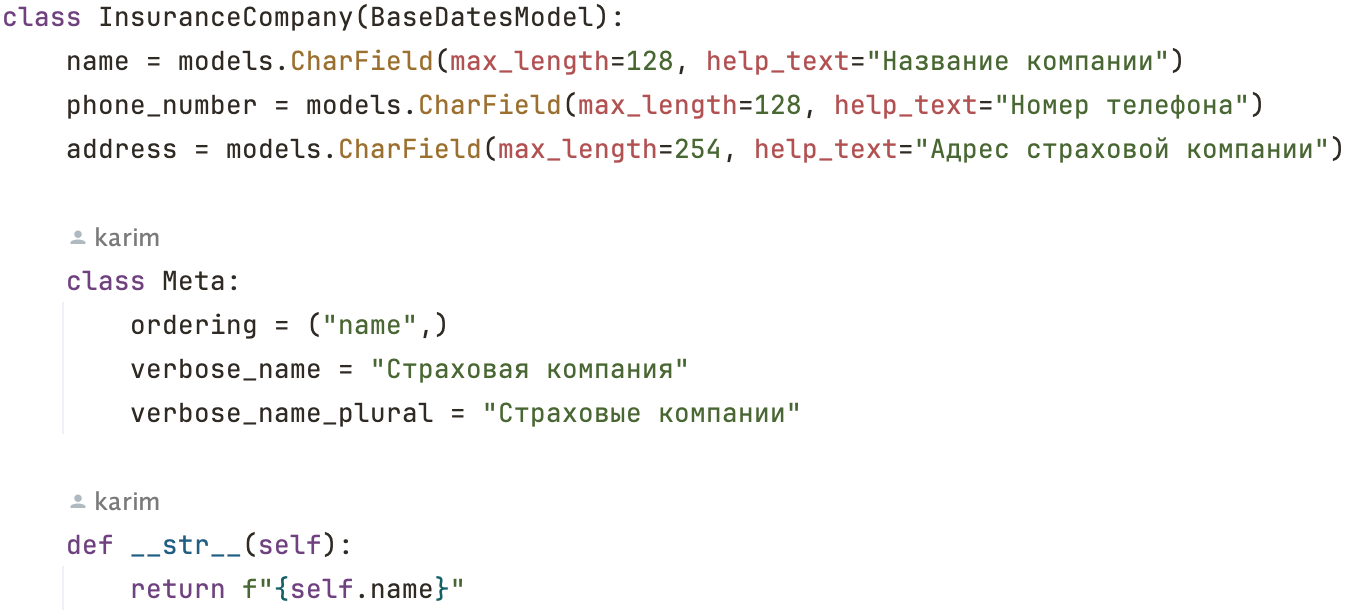


Рисунок 5 – Процесс создания базы данных с помощью Django

Чтобы наполнить БД данными в Django можно использовать «фикстуры». Фикстуры представляют собой способ загрузки начальных данных в базу данных при развертывании или обновлении приложения. Они позволяют определить набор предварительно подготовленных данных, которые будут автоматически загружены в базу данных.

Фикстуры в Django могут быть представлены в различных форматах, таких как JSON, XML, YAML или CSV. Каждая фикстура обычно содержит набор объектов определенной модели, которые будут созданы и сохранены в базе данных.

Одним из основных преимуществ использования фикстур является возможность создания предопределенного набора данных для тестирования или инициализации приложения.

На рисунках 6 представлены фикстуры для модели врачей.



Рисунок 6 – JSON файл с фикстурами

Чтобы загрузить подготовленные фикстуры в БД нужно использовать команду loaddata [путь к файлу с фикстурами]

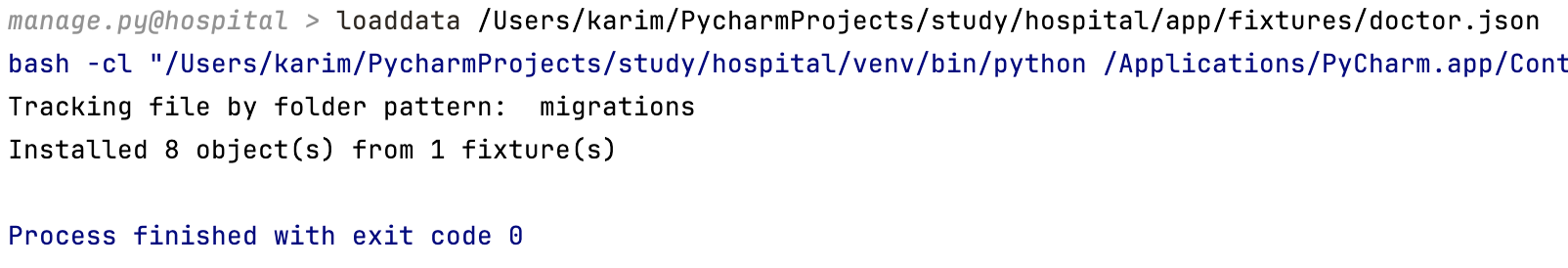


Рисунок 7 – Загрузка фикстур

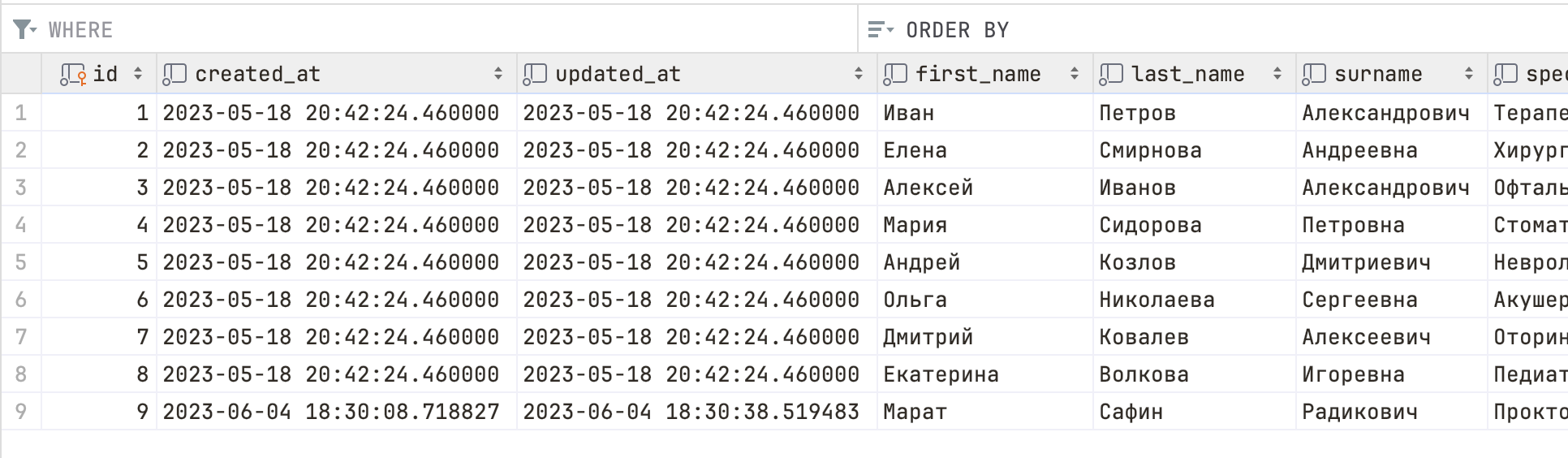


Рисунок 8 – Наполненная база данных

# 6 Создание клиентской части ИС

При запуске приложения пользователь таблицу с докторами. Таблицу с докторами можно увидеть на рисунке 9.

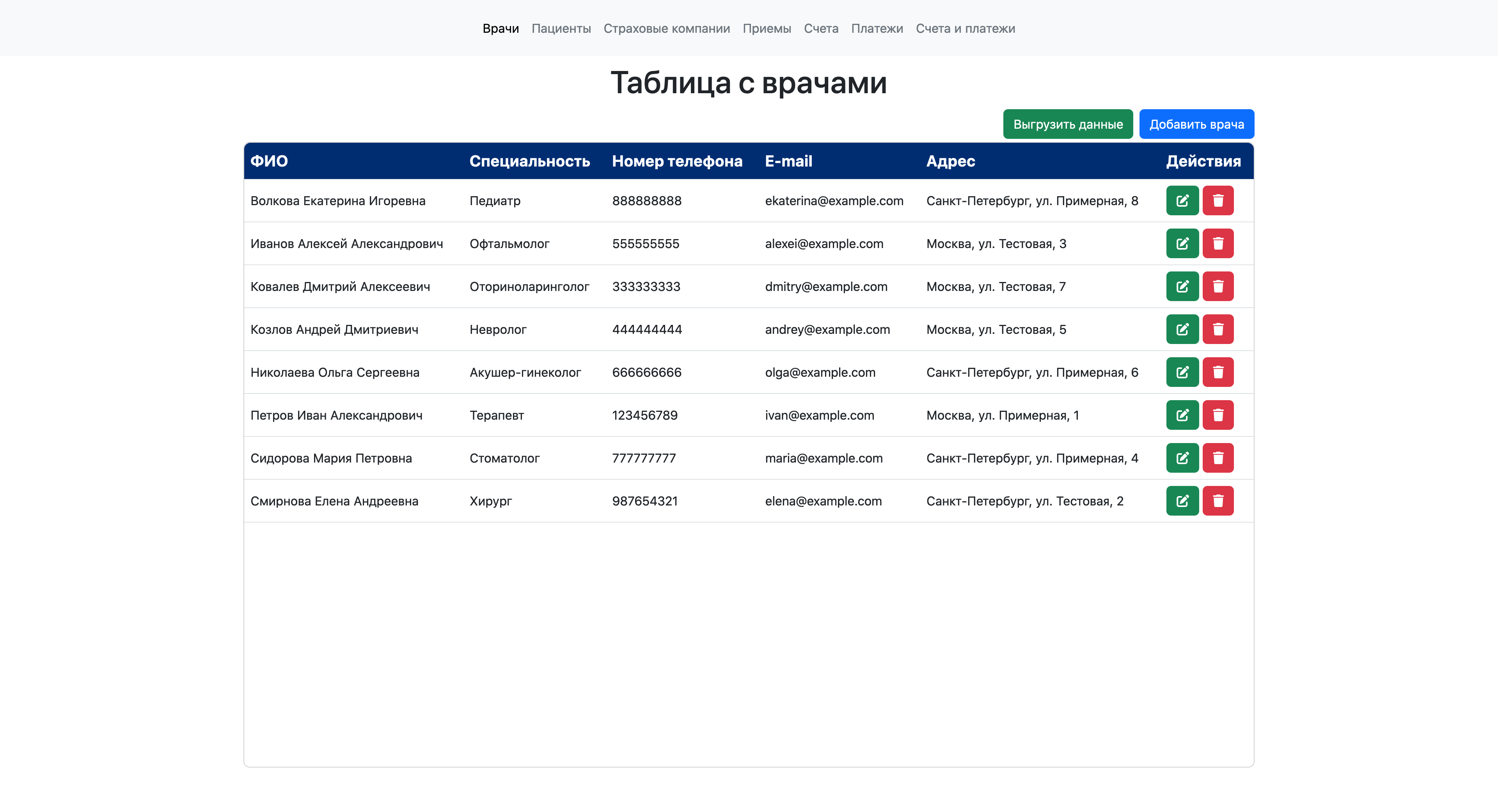


Рисунок 9 – Таблица с врачами

В верхней части страницы можно увидеть меню с названиями других таблиц. Далее отображается название таблицы и действия, которые можно сделать (добавить врача и выгрузить данные из таблицы).

После нажатия на кнопку «Добавить врача» пользователь будет перенаправлен на таблицу с формой для заполнения данных о враче. Форма представлена на рисунке 10.

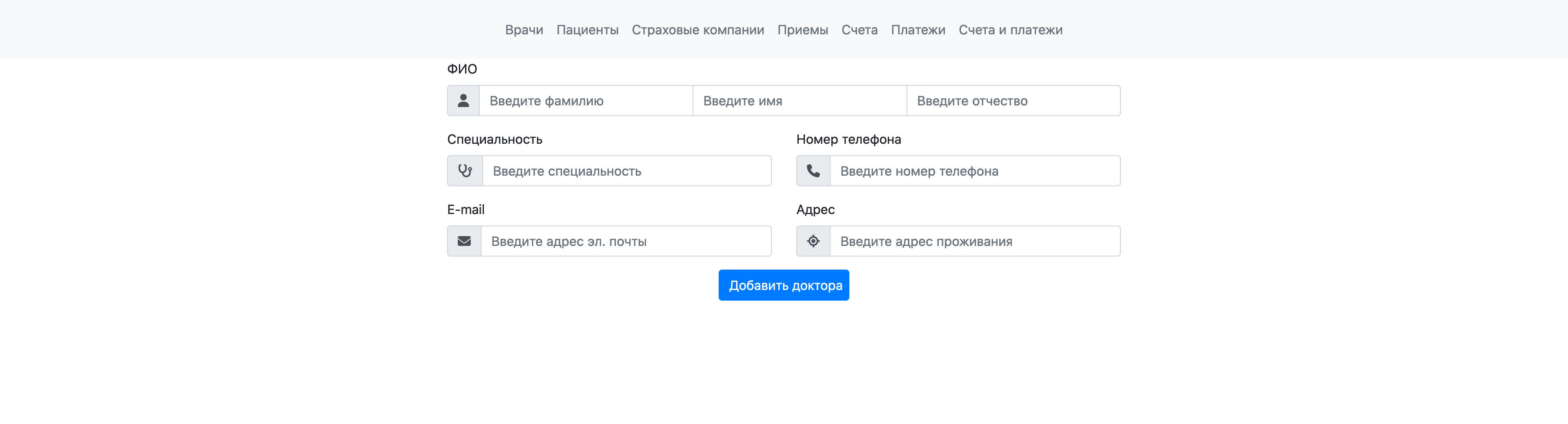
****

Рисунок 10 – Форма для добавления врача

Если пользователь не заполнит обязательное поле, будет выведена соответствующая ошибка, что и представлено на рисунке 11.

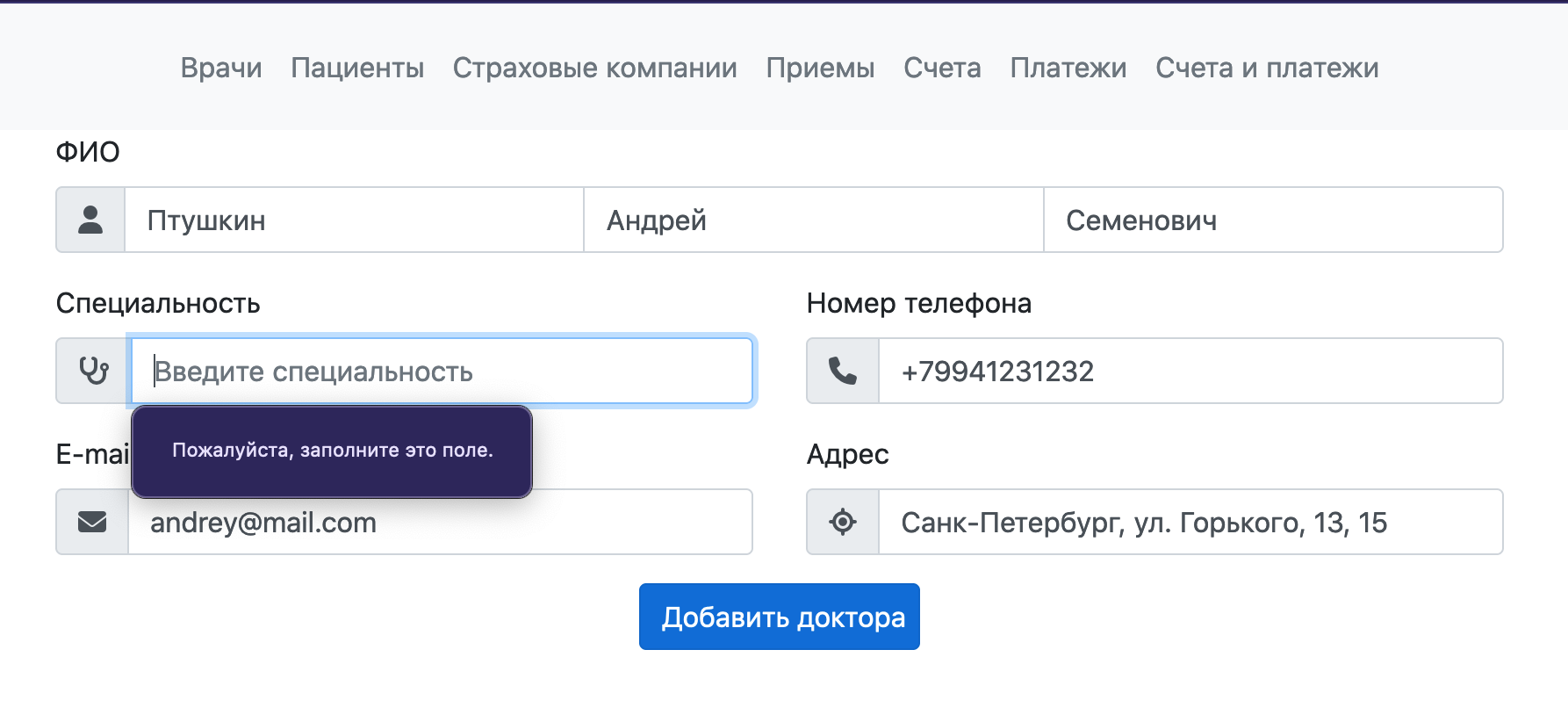


Рисунок 11 – Ошибка при неверном заполнении формы

После добавления врача в таблицу, в таблице появляется соответствующая запись. Обновленная таблица представлена на рисунке 12.

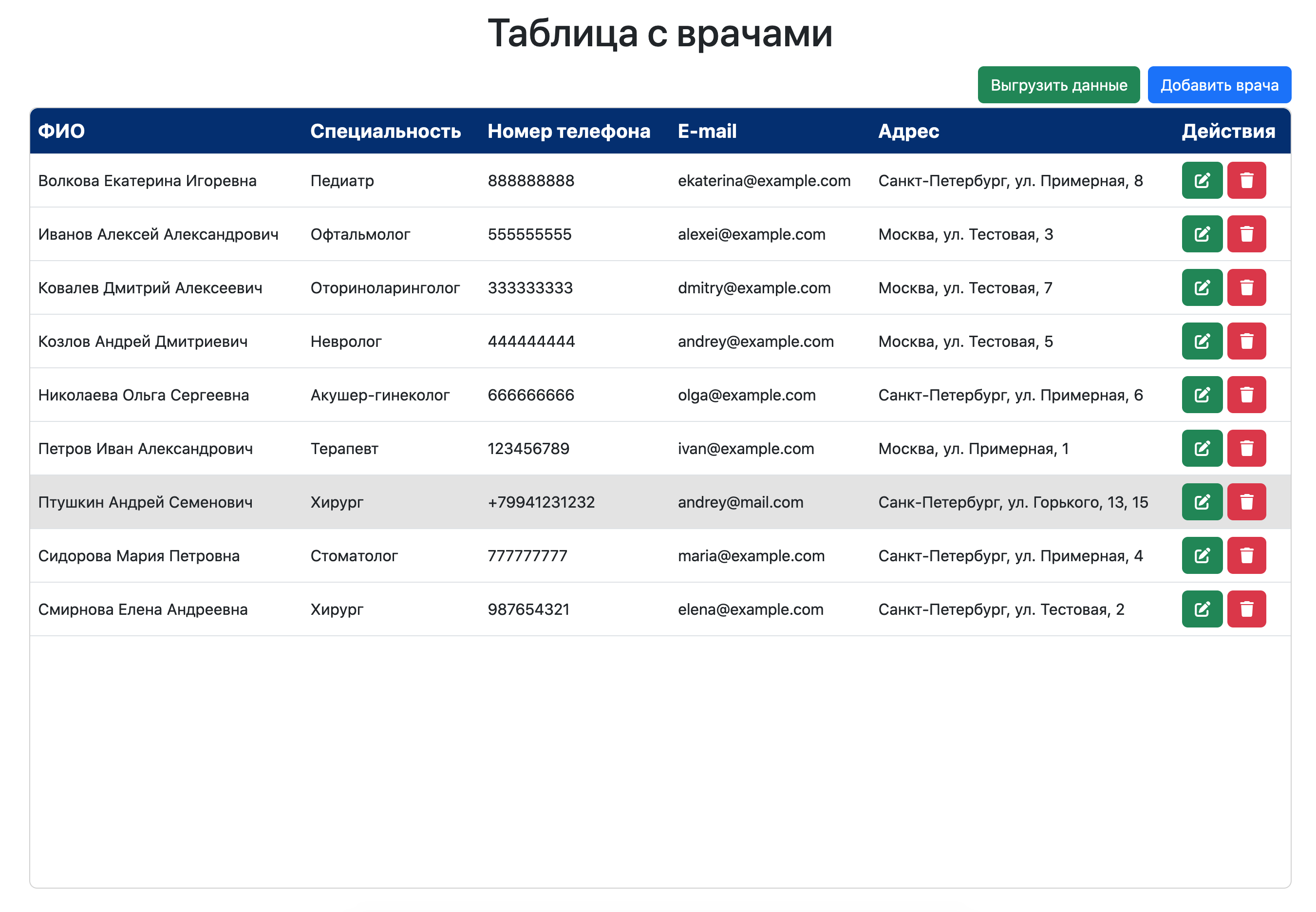


Рисунок 12 – Добавление записи в таблицу

При нажатии на кнопку «Выгрузить данные» предлагается выбрать путь сохранения для .xlsx файла c данными из таблицы, после чего начинается загрузка, что и представлено на рисунке 13.

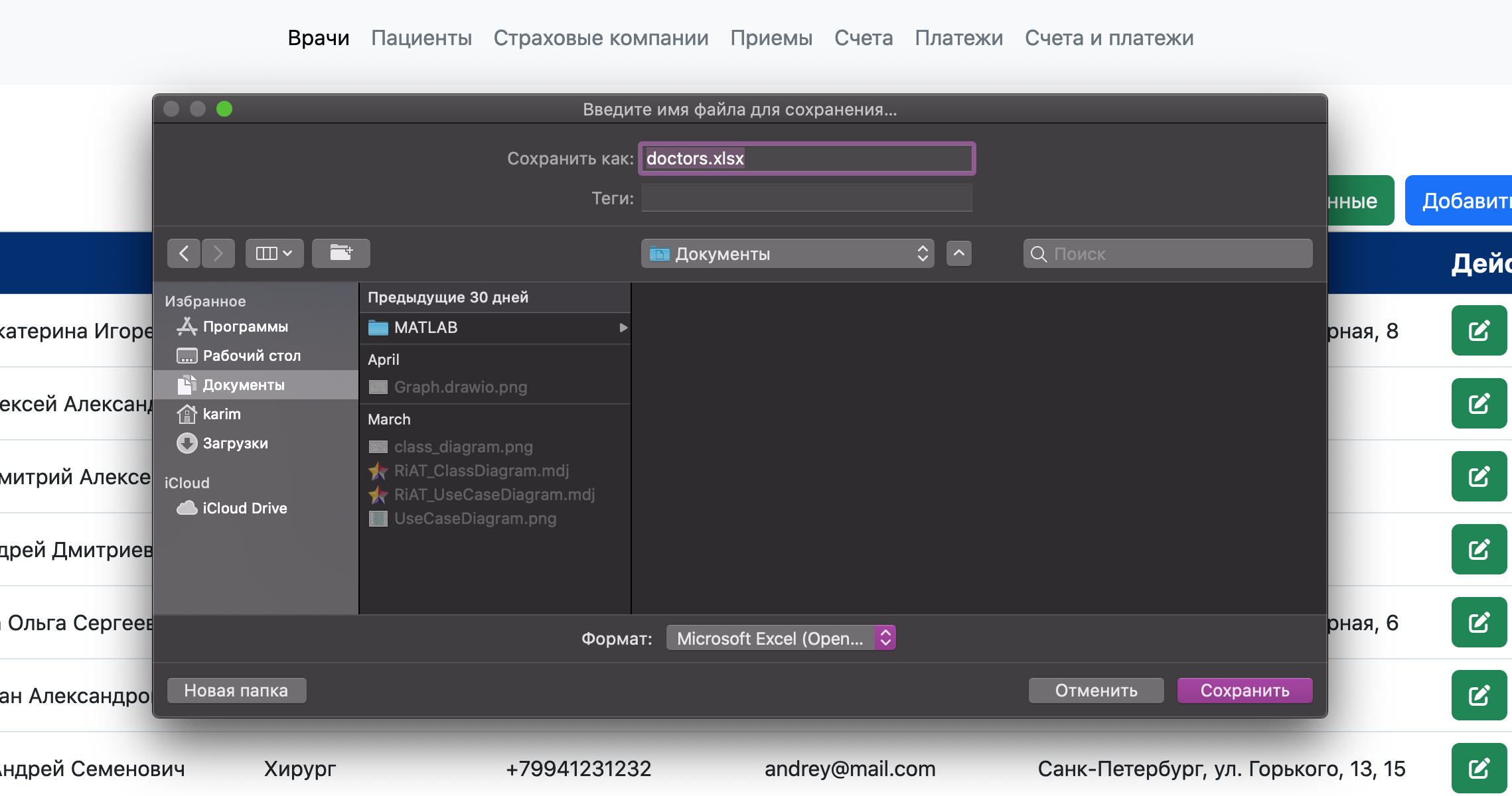


Рисунок 13 – Экспорт данных из таблицы

Содержание загруженного файла представлено на рисунке 14.

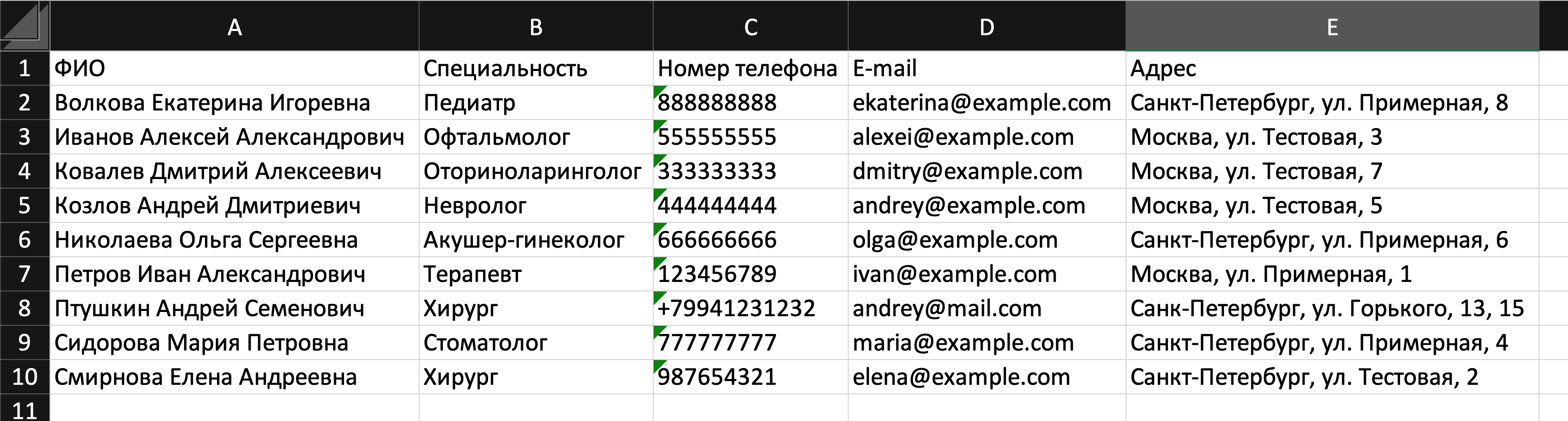


Рисунок 14 – Содержание загруженного .xlsx файла

Также рядом с каждой записью можно увидеть кнопки с действиями, зеленая отвечает за редактирование записи, красная – за удаление.

При нажатии на кнопку редактирования пользователь будет перенаправлен на страницу с заполненной формой врача, в которой он сможет отредактировать данные. На рисунке 15 продемонстрирована заполненная форма. После нажатия на кнопку «Обновить врача» соответствующая запись в таблице будет обновлена.

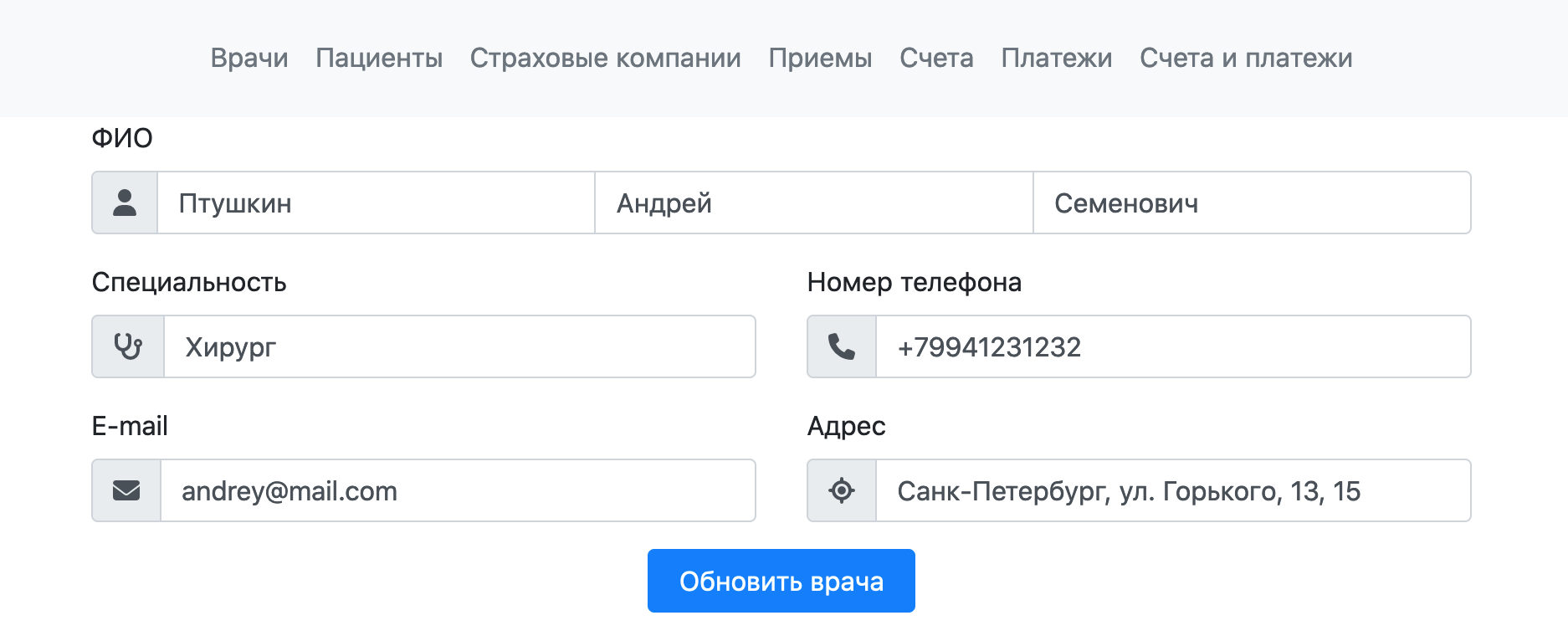


Рисунок 15 – Форма редактирования врача

После нажатия на кнопку «Обновить врача» изменённая информация будет отображена в базе данных и в соответствующих таблицах. На рисунке 16 представлена измененная запись станций.

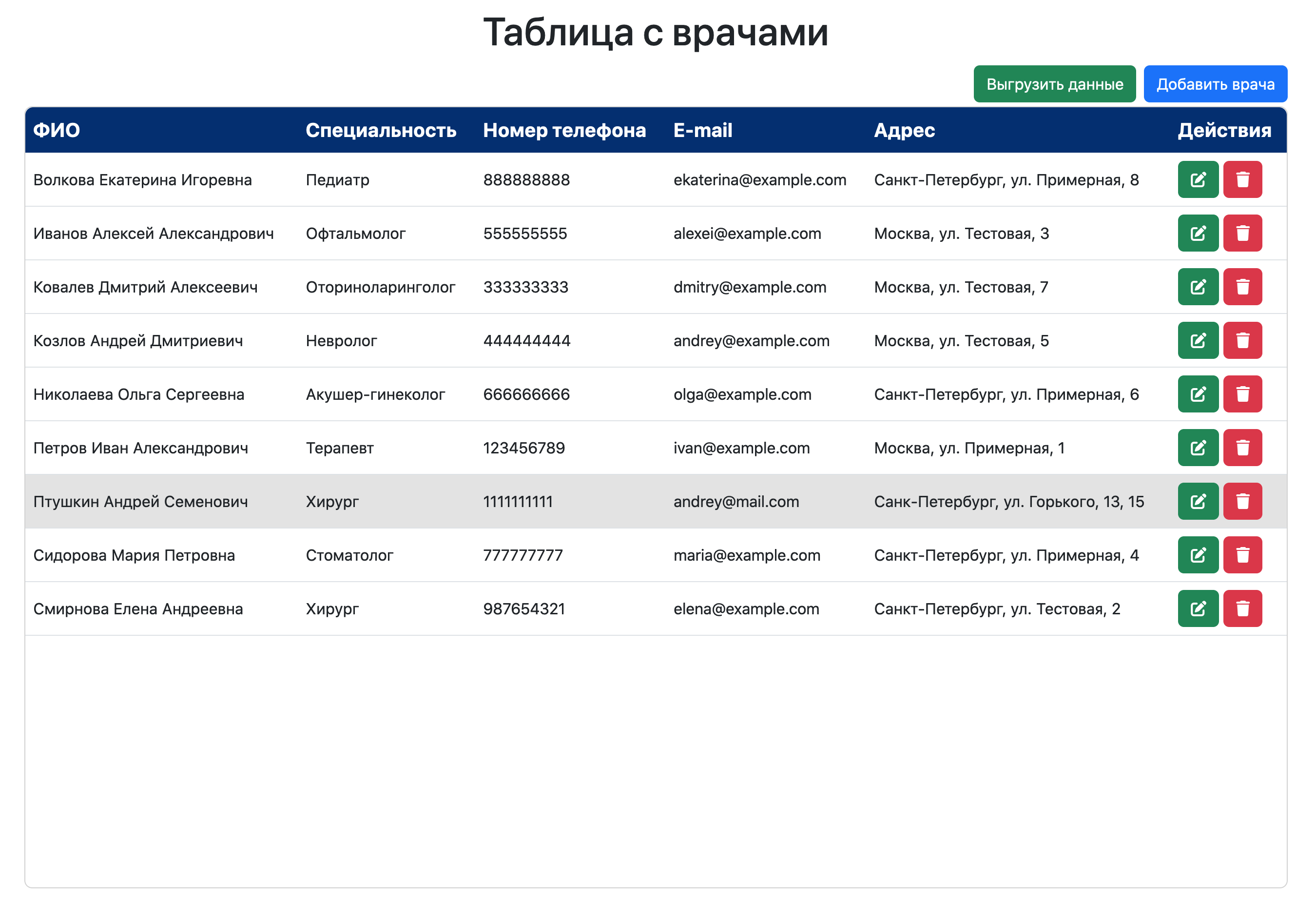


Рисунок 16 – Измененная информация о враче

Далее представлен пример связанной таблицы, в качестве примера взята таблица связи приемов, платежей и счетов.

На рисунке 17 представлена таблица с счетами и платежами. Кнопка «Выгрузить данные» работает аналогично с остальными таблицами.

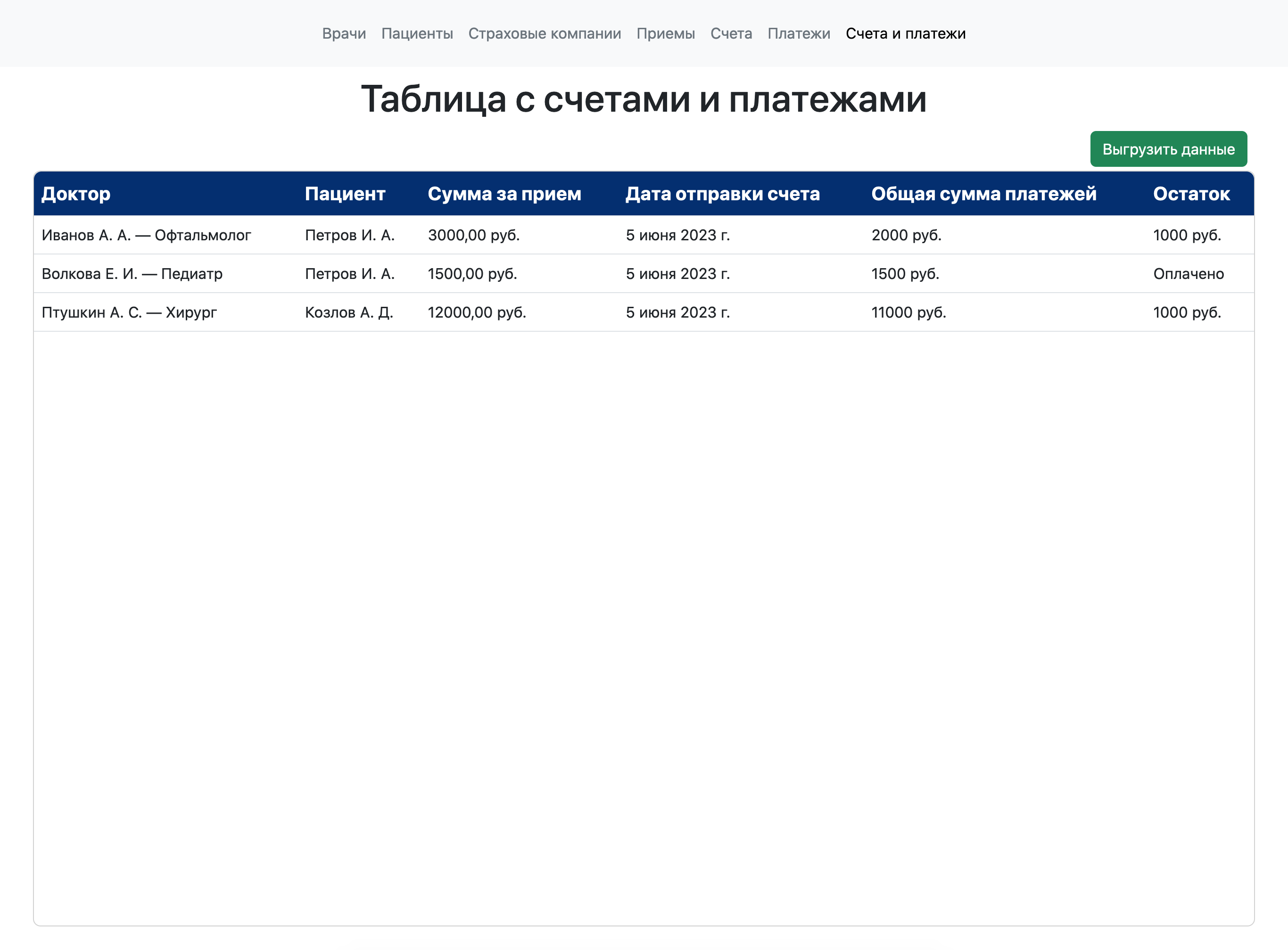


Рисунок 17 – Таблица счетов и платежей

Колонка «Общая сумма платежей» формируется путем складывания всех платежей, внесенных в счет, в этом можно убедиться, посмотрев таблицу «Платежи», продемонстрированную на рисунке 18.

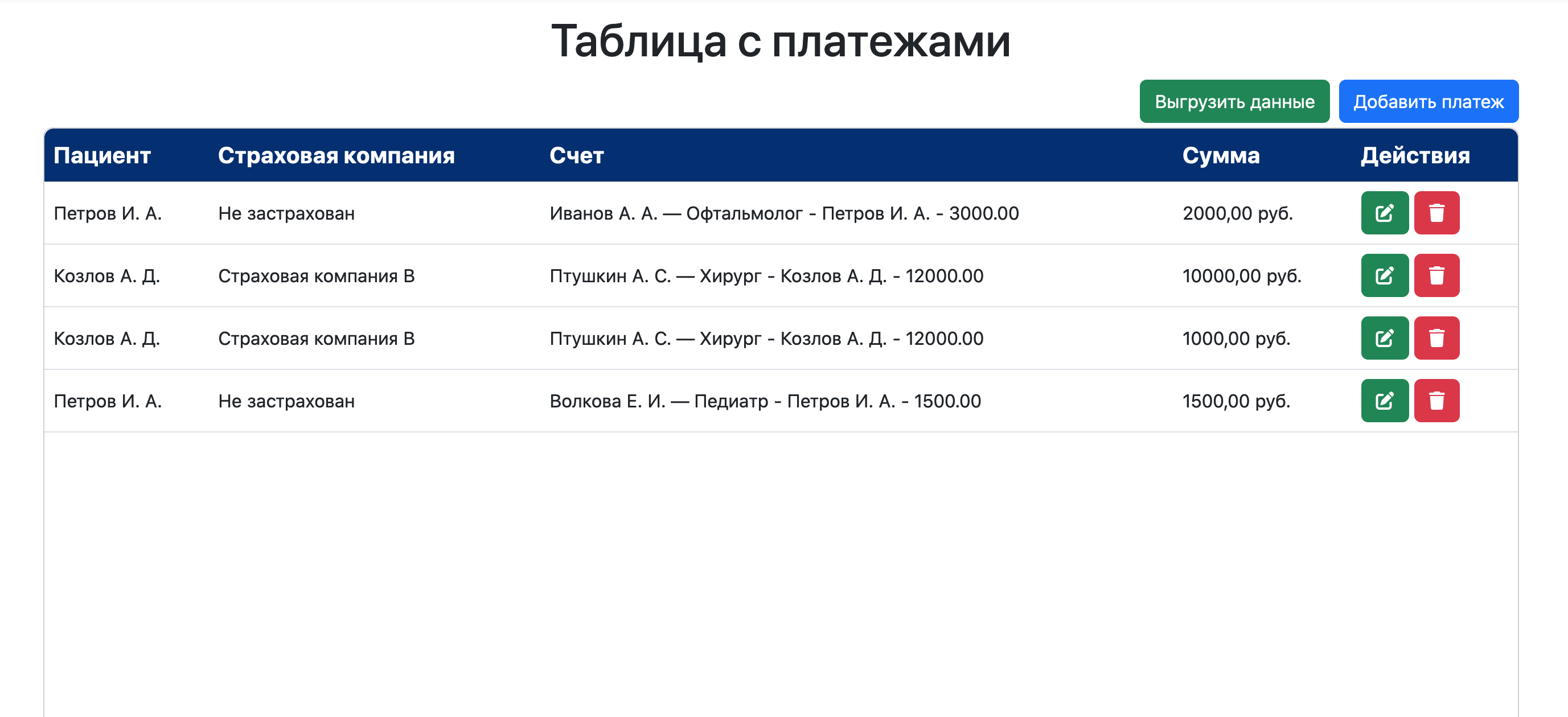


Рисунок 18 – Таблица с платежами

В таблице видно, что пациент Козлов А. Д. вносил 2 платежа, которые в сумме равны 11000 руб. Если мы нажмем на кнопку «Добавить платеж», появится форма, показанная на рисунке 19.

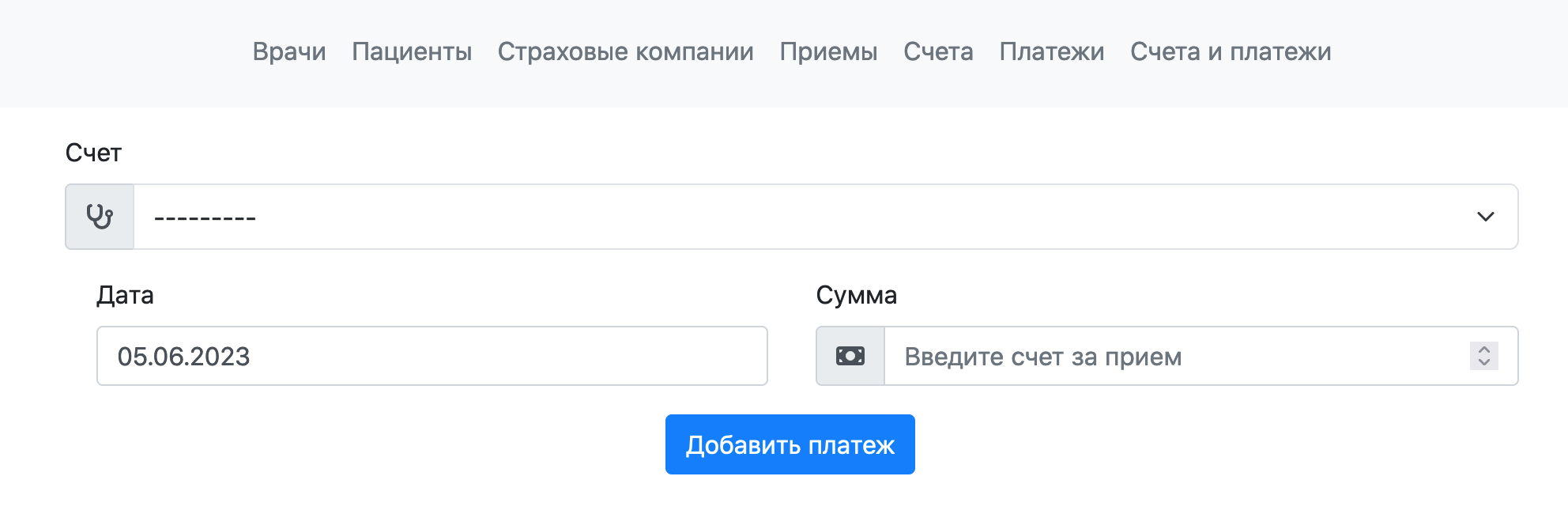


Рисунок 19 – Добавление платежа

В форме можно увидеть только 3 поля для заполнения, вместо 5, что есть в модели таблицы «Платежа». После выбора счета из выпадающего списка, введения даты и суммы платежа и нажатия на кнопку «Добавить платеж» в системе вычисляется принадлежит ли пациент к страховой компании или же нет. Если выяснилось, что пациент застрахован, в поле «пациент» записывается «None», а полю «страховая компания» присваивается соответствующий идентификатор страховой компании. Заполненную форму можно увидеть на рисунке 20.

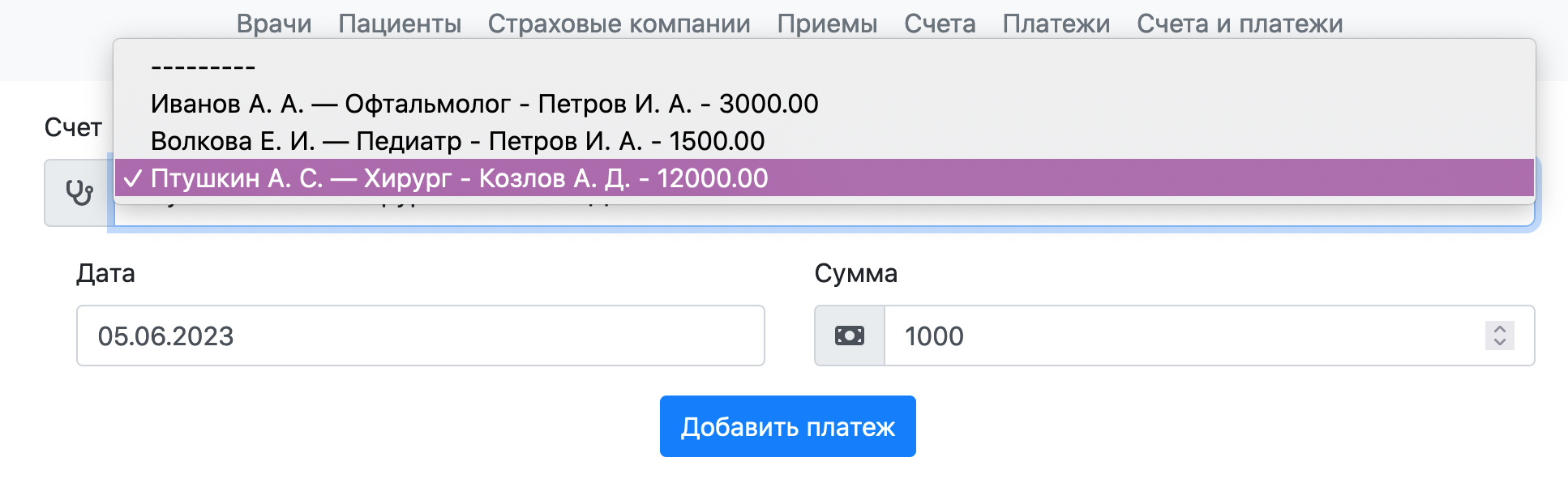


Рисунок 20 – Заполненная форма с платежом

На рисунке 21 показана добавленная запись в таблицу.

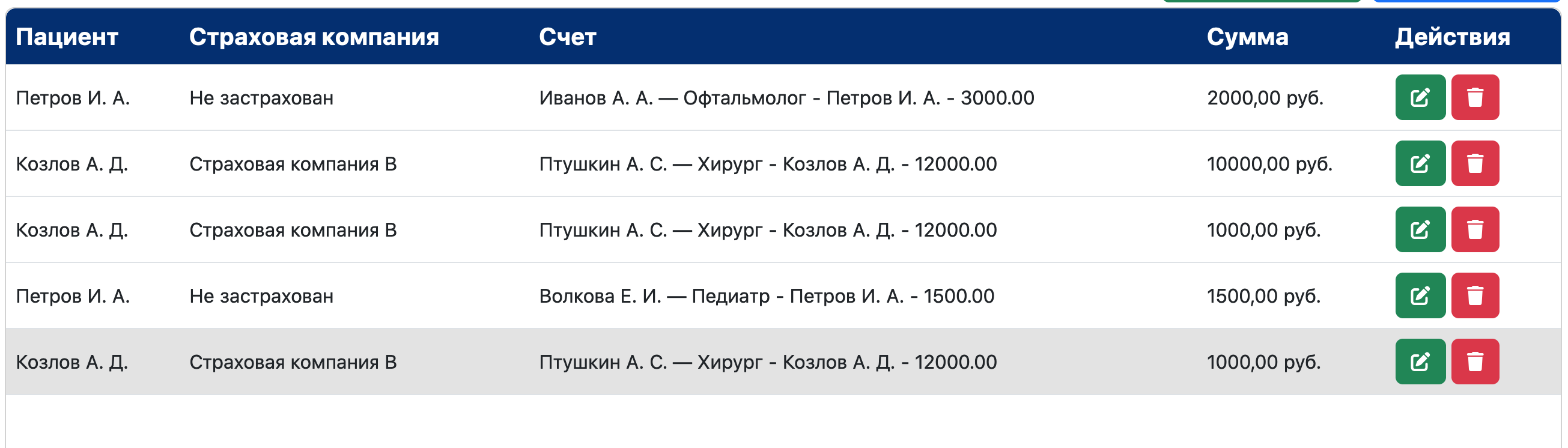


Рисунок 21 – Добавленная запись с платежом в таблицу.

Теперь в таблице с счетами и платежами можно увидеть, что счет за прием Козлова А. Д. у хирурга оплачен. Обновленную таблицу с счетами и платежами можно увидеть на рисунке 22.



Рисунок 22 – Обновленная страница с счетами и платежами

При попытке удалить запись, пользователь будет перенаправлен на страницу с подтверждением. Страница представлена на рисунке 23.

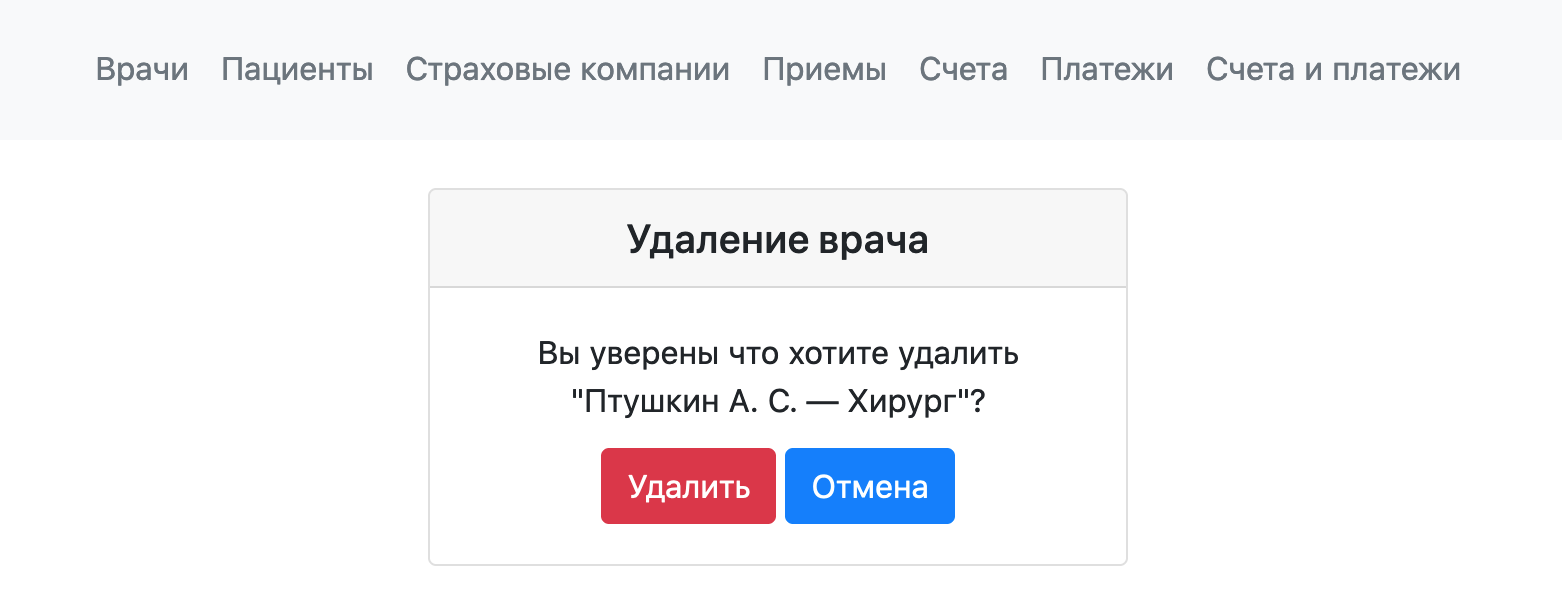


Рисунок 23 – Удаление записи

Далее рассмотрена форма записи на прием. Чтобы записать пациента к врачу сотруднику поликлиники сперва нужно выбрать пациента. Затем нужно выбрать специальность доктора, к которому пациент хочет записаться на прием. После выбора специальности предоставляется список для выбора врача соответствующей специальности, что и представлено на рисунке 24.

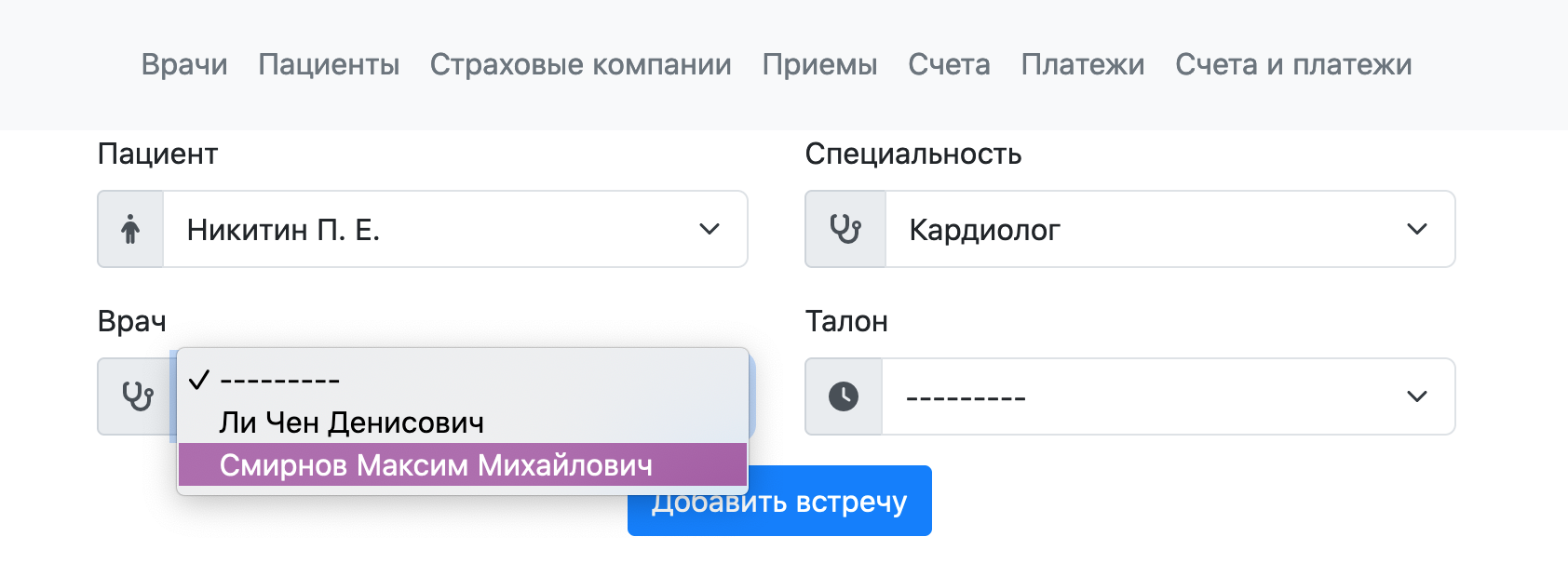


Рисунок 24 – Запись на прием к врачу (выбор доктора и специальности)

После выбора доктора в поле с выбором талона будут доступны свободные талоны к соответствующему врачу. Если раннее другой пациент был записан, этот талон больше не будет отображаться в списке доступных. Список доступных талонов показан на рисунке 25.

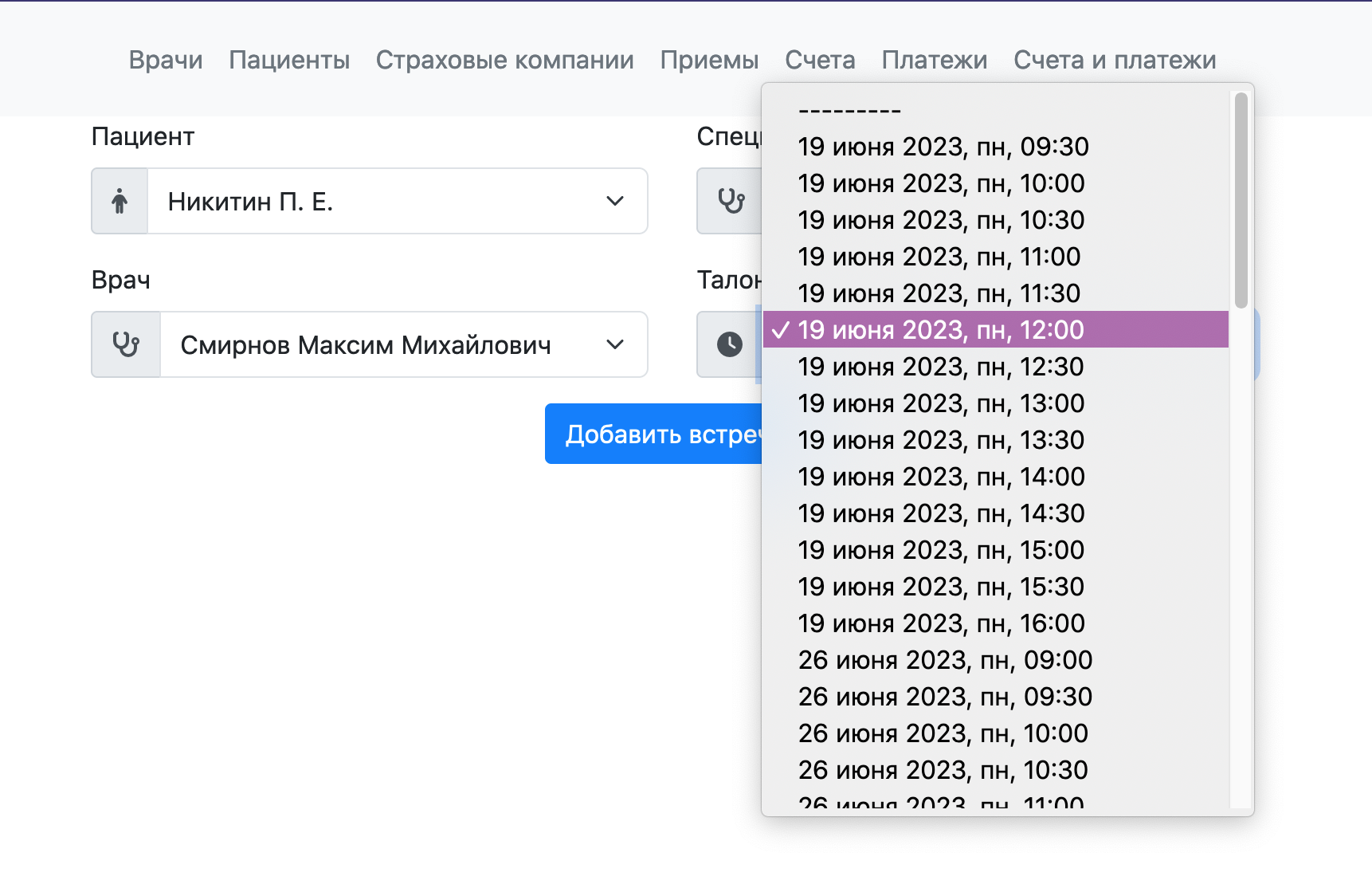


Рисунок 25 – Список доступных талонов

Добавление талонов происходит автоматически после добавления расписания на день. Это реализовано с помощью инструмента Django signal. Он позволяет отлавливать события Pre\_save, Post\_save, Pre\_delete, Post\_delete. В данном случае был выбран сигнал Post\_save таблицы schedule. Т. е. после добавления расписания запускается функция, генерирующая талоны с интервалом в 30 минут на четыре недели вперед. Эта функция представлена на рисунке 26.

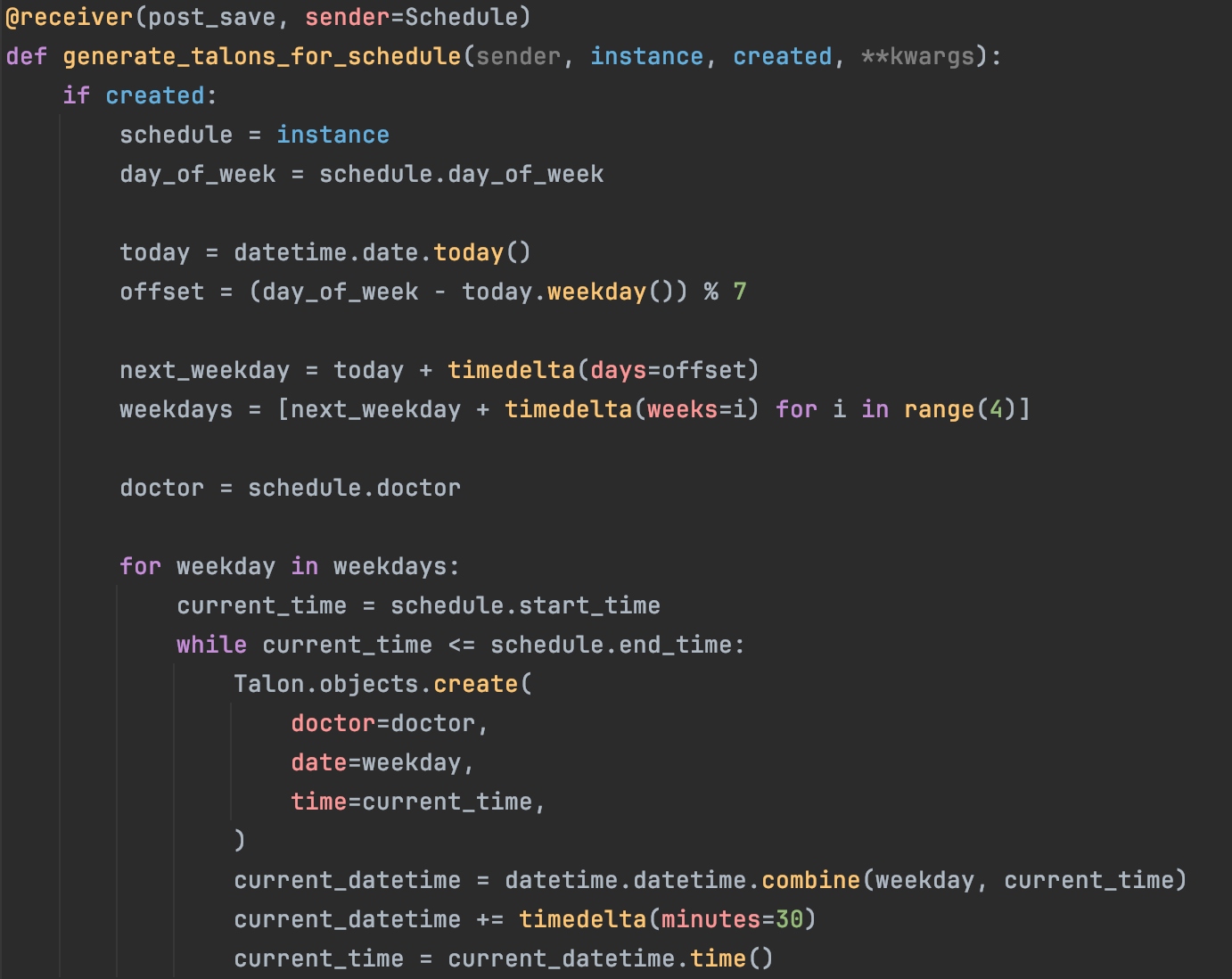


Рисунок 26 – Функция, генерирующая талоны

Запросы к БД осуществляются с помощью Django ORM как было сказано выше. Далее будет разобран запрос к БД на примере таблицы «Счета и платежи». По сути, данный запрос направлен на получение информации о счетах (Bill) с определенными аннотациями и связанными данными. Вот описание запроса:

1. В начале мы выбираем все объекты модели Bill с использованием метода «***select\_related***», чтобы выполнить присоединение (***join***) с моделью Appointment и связанной с ней моделью Doctor.
2. Затем мы используем аннотацию «***total\_payment***», которая вычисляет общую сумму платежей для каждого счета. Для этого мы используем подзапрос «***total\_payment\_subquery***», который возвращает общую сумму платежей, связанных с каждым счетом. Результат этого подзапроса сохраняется в поле «***total\_payment***».
3. Далее мы аннотируем поле «***balance***», которое вычисляет остаток (разницу) между суммой счета (***amount***) и общей суммой платежей (***total\_payment***).
4. Также мы аннотируем поле «***doctor***», которое объединяет несколько полей модели Doctor для представления имени доктора в формате "Фамилия И. О. — Специальность". Это достигается с помощью функции «***Concat***», которая объединяет значения полей с заданными разделителями.
5. Наконец, мы аннотируем поле «***patient***», которое объединяет несколько полей модели Patient для представления имени пациента в формате "Фамилия И. О.".

В результате выполнения данного запроса получается QuerySet, который содержит объекты модели Bill с дополнительными аннотациями «***total\_payment***», «***balancе***», «***doctor***» и «***patient***», предоставляя информацию о счетах, общей сумме платежей, остатке, а также связанных данными о докторе и пациенте.



Рисунок 27 – Пример сложного запроса

# ЗАКЛЮЧЕНИЕ

Конечные результаты курсового проекта включают в себя следующие выполненные задачи:

1. Изучение и описание предметной области.
2. Разработка инфологической модели данных предметной области.
3. Преобразование инфологической модели в даталогическую модель данных.
4. Обеспечение перевода даталогической модели в среду СУБД MySQL.
5. Наполнение разработанной базы данных данными.
6. Составление необходимых запросов к БД для дальнейшего использования в клиентской части ИС.
7. Реализация пользовательского интерфейса.
8. Тестирование работоспособности программы.

В результате успешного выполнения всех указанных задач было разработано ИПК для полиники. В связи с этим, можно считать, что все задачи были достигнуты, цель курсового проекта выполнена.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. База данных. Понятие, значение и роль в современном мире [Электронный ресурс] – URL: https://cyberleninka.ru/article/n/baza-dannyh-ponyatie-znachenie-i-rol-v-sovremennom-mire/viewer (дата обращения: 30.05.2023).
2. Лекция 7: Инфологическое моделирование [Электронный ресурс] – URL: https://intuit.ru/studies/courses/1001/297/lecture/7411 (дата обращения: 23.05.2023).
3. MySQL Documentation [Электронный ресурс] – URL: https://dev.mysql.com/doc/ (дата обращения: 23.05.2023).
4. Руководство по языку программирования Java [Электронный ресурс] – URL: https://metanit.com/python/tutorial/ (дата обращения: 23.05.2023).
5. Django Documentation [Электронный ресурс]. – URL: https://docs.djangoproject.com/en/3.2/ (дата обращения: 22.05.2023).

# ПРИЛОЖЕНИЕ А

**Исходный код программы**

Исходный код находится по следующему URL: https://github.com/karim3487/db\_course\_work