Дипломная работа по теме: Разработка веб-приложения - Портал приема заявок и записи на консультации   
  
Автор: Дегтярев Артем Валерьевич

Оглавление

[1. Введение 1](#_Toc183598175)

[2.1 Django 3](#_Toc183598176)

[2.2 Flask 4](#_Toc183598177)

[2.3 FastAPI 4](#_Toc183598178)

[2.4 Выбор для моего приложения 5](#_Toc183598179)

[Разработка ключевого функционала 6](#_Toc183598180)

[Основные требования 7](#_Toc183598181)

[4. Разработка в соответствии с созданной документацией 7](#_Toc183598182)

[5.Заключение 13](#_Toc183598183)

[Приложение 1. Список необходимых библиотек 14](#_Toc183598184)

1. Введение  
  
Обоснование выбора темы:  
  
1. В современном мире борьба за каждого клиента в области продаж в том числе и в моей профессиональной сфере электроэнергетики важна и требует постоянного повышения доступности к услугам, особенно в сфере консультаций. Упрощение процессов с целью повышения скорости вовлечения новых клиентов путем автоматизации записи на консультации и получения сведений о клиентах играют ключевую роль в увеличении продаж, а также снижает нагрузку на сотрудников и минимизирует вероятность ошибок при управлении расписанием.  
  
  
2. Потребности рынка. Компании стремятся использовать наиболее эффективные модели для улучшения качества предлагаемых ими услуг. Разработанное приложение, направленное на получение информации о клиентах и оказания услуг по консультациям, будет всегда востребована в любых отраслях бизнеса.

3. Практическая значимость: Работа в области разработки веб приложения позволит применить полученные знания на практике и даст опыт использования приложения для дальнейшего внедрения его на сайт моей компании. Это позволит получать информацию о клиентах для базы продаж и планировать с ними встречи в запланированное ими время, что повысит удовлетворенность клиентов.  
  
4. Личный интерес и потенциальные перспективы: Использование портала приема заявок и записи на консультации является внедрением уже в реальном сайте и возможности опробования его на сайте своей компании, что в дальнейшим послужит базой для улучшения, внедрения новых инструментов для увеличения продаж и повышения уровня клиентоориентированности путем предоставлений индивидуальных консультаций.   
  
Таким образом, выбор темы Разработка веб-приложения - Портал приема заявок и записи на консультации." обусловлен ее актуальностью, потребностями рынка, практической значимостью и личным интересом, что делает эту тему подходящей для проведения дипломной работы.  
  
Определение цели и задачи:  
  
Цель: Разработать веб-приложение, которое позволит пользователям отправлять заявки на консультации и записываться на встречи в календаре. Также необходимо реализовать функционал отправки SMS-напоминаний о запланированных встречах.   
  
Задачи:  
  
1. Обзор фреймворков для написания веб приложений, применяющихся для разработки веб-приложений. Провести анализ существующих инструментов, выбрать наиболее подходящие под имеющуюся задачу.  
  
2. Выполнить регистрацию и аутентификацию пользователей: - Пользователи могут создавать учетные записи и входить в систему. Личный кабинет для просмотра статуса заявок и запланированных встреч.

3. Создать форму для заполнения заявки: - Пользователи заполняют форму с их данными, включая имя, контактную информацию, описывают продукт, по которому нужна консультация. Пользователи могут выбирать дату и время для встречи в календаре. Заявки сохраняются в базе данных.

4. Интеграция с API на базе библиотеки Django Rest Framework

4. Отправка SMS-напоминаний: - SMS-услуги для отправки напоминаний о встречах. - Напоминания отправляются сразу для заблаговременного планирования клиентом планов

5. Административная панель: - Администраторы могут просматривать все заявки, менять их статус, редактировать информацию о пользователях или встречах. - Возможность удаления ненужных записей.  
  
6. Написание дипломной работы: составить дипломную работу, включающую в себя введение, обзор литературы, методологию и результаты исследования, анализ результатов, выводы и рекомендации.  
  
Цели и задачи исследования направлены на получение практических результатов, которые позволят увеличить клиентскую базу и повысить клиент-ориентированность для моей компании.

2. Обзор популярных инструментов для разработки веб-приложений на Python

Python предлагает множество фреймворков для веб-разработки, среди которых наиболее известными являются Django, Flask и FastAPI. Каждый из этих инструментов имеет свои особенности и предназначение, что позволяет разработчикам выбирать наиболее подходящий фреймворк в зависимости от специфики проекта. Давайте рассмотрим каждый из них более подробно.

2.1 Django — это мощный и высокоуровневый фреймворк для веб-разработки на Python, созданный для упрощения процесса создания сложных веб-приложений.

Основные возможности:

- Встроенная страница администрирования: позволяет быстро управлять данными приложения без необходимости создания отдельной панели.

- ORM (Object-Relational Mapping): облегчает взаимодействие с базами данных, позволяя разработчикам работать с данными, не прибегая к написанию SQL-кода.

- Поддержка аутентификации и управления доступом: Инструменты для реализации систем регистрации, входа в систему и контроля доступа к различным частям приложения.

- Мощная система маршрутизации URL: обеспечивает удобное управление маршрутами и обработчиками запросов.

- Шаблонизатор: позволяет генерировать динамические HTML-страницы с помощью шаблонов.

Особенности:

- Django идеально подходит для крупных и сложных проектов благодаря своей развитой экосистеме, поддерживающей множество дополнительных приложений и библиотек.

- Высокая структура и следование принципам проектирования делают его отличным выбором для долгосрочных проектов с командной разработкой.

2.2 Flask — это легковесный микро-фреймворк для веб-разработки на Python, предоставляющий минималистичный подход к созданию приложений.

Основные возможности:

- Минималистичный подход: Flask предоставляет только основные компоненты, что позволяет избежать лишних зависимостей и фреймворк-функций.

- Высокая гибкость: Разработчики могут добавлять необходимые библиотеки и компоненты, создавая приложение, которое идеально соответствует их требованиям.

- Использование Jinja2 и Werkzeug: Jinja2 обеспечивает мощный движок шаблонов, а Werkzeug — обширную библиотеку для маршрутизации и обработки HTTP-запросов.

Особенности:

- Flask хорошо подходит для небольших и средних проектов, где требуется высокая степень настройки и гибкости. Его простота делает его идеальным для быстрого прототипирования.

2.3 FastAPI — это современный, высокопроизводительный фреймворк, предназначенный для создания API, основанный на стандартных спецификациях OpenAPI и JSON Schema.

Основные возможности:

- Высокая производительность: использование асинхронного программирования (async/await) обеспечивает быстрое выполнение запросов, что делает его отличным выбором для API.

- Автоматическая генерация документации: FastAPI автоматически генерирует интерактивную документацию к API, что упрощает его использование и интеграцию.

- Поддержка типизации: позволяет валидировать данные и делает код более читаемым и безопасным.

- Интеграция с библиотеками: Удобная работа с библиотеками, такими как Pydantic и SQLAlchemy, для упрощения работы с данными.

Особенности:

- FastAPI часто используется для разработки API и микросервисов, что делает его идеальным для современных облачных приложений и серверных решений.

## 2.4 Выбор для моего приложения

Итог: при выборе стека технологий для моего веб-приложения я остановился на Django, и на это есть несколько причин:

* Полнота функционала: Django предлагает множество встроенных возможностей, таких как аутентификация, администрирование и ORM, что позволяет нам сосредоточиться на логике приложения, а не на создании базовых компонентов.
* Структурированность проекта: благодаря своей структуре и хорошо определенным принципам проектирования Django способствует созданию масштабируемых и поддерживаемых приложений. Это особенно актуально, учитывая долгосрочные цели нашего проекта.
* Большое сообщество и поддержка: Django имеет огромное сообщество и большое количество доступных ресурсов, библиотек и плагинов, что облегчает поиск помощи и внедрение новых функций.
* Безопасность: Django предоставляет ряд встроенных инструментов для защиты приложений, включая защиту от атак SQL-инъекций и CSRF. Это критически важно для обеспечения безопасности данных пользователей.
* Готовность к масштабированию: если в будущем потребуется добавить новые функции или расширить приложение, Django может легко адаптироваться к этим изменениям благодаря своей архитектуре и возможностям.

В совокупности эти факторы делают Django моим выбором для разработки данного веб-приложения, обеспечивая не только надежность и производительность, но и достаточную гибкость для реализации всех необходимых функций.  
  
  
3. Проектирование приложения  
  
Планирование и анализ требований  
  
Выбор фреймворка и инструментов: Определение наиболее подходящего фреймворка (выбор сделан на Django) и инструментов для разработки.  
  
Определение структуры приложения: Разработка схемы архитектуры, включающей фронтенд, бэкенд, базу данных и интеграцию с API.

Разработка ключевого функционала  
  
Выполнить регистрацию и аутентификацию пользователей: пользователи могут создавать учетные записи и входить в систему. Личный кабинет для просмотра статуса заявок и запланированных встреч.

Создать форму для заполнения заявки: пользователи заполняют форму с их данными, включая имя, контактную информацию, описывают продукт, по которому нужна консультация. Пользователи могут выбирать дату и время для встречи в календаре. Заявки сохраняются в базе данных.

Выполнить интеграцию с API на базе библиотеки Django Rest Framework

Отправка SMS-напоминаний: SMS-услуги для отправки напоминаний о встречах. Напоминания отправляются сразу для заблаговременного планирования клиентом планов. После покупки компанией платного номера подключить уже API.

Административная панель: администраторы могут просматривать все заявки, менять их статус, редактировать информацию о пользователях или встречах. Возможность удаления ненужных записей.  
  
Реализация пользовательского интерфейса: Создание форм и страниц для веб-приложения, отображения результатов и взаимодействия с пользователем.

Основные требования  
  
Пользователь должен иметь возможность регистрации, записи в календаре на консультацию с уведомлением о событии посредством СМС, а также иметь личный кабинет с информацией о всех его заявках.  
  
  
Технические требования  
  
Фронтенд: использовать HTML, CSS, JavaScript, Bootstrap  
  
Бэкенд: Python с использованием фреймворков Django.  
  
База данных: Использование базы данных (SQLite) для хранения данных о пользователях и истории загрузок.  
  
API: Разработка API для взаимодействия между фронтендом и бэкендом.

4. Разработка в соответствии с созданной документацией  
  
Планирование разработки  
  
Разработка была разделена на основные этапы: проектирование интерфейса, реализация серверной логики.  
  
**Выполнена установка необходимых библиотек, создано виртуальное окружение, проект Django и приложения, а также создана необходимая структура проекта (Рис.1).**

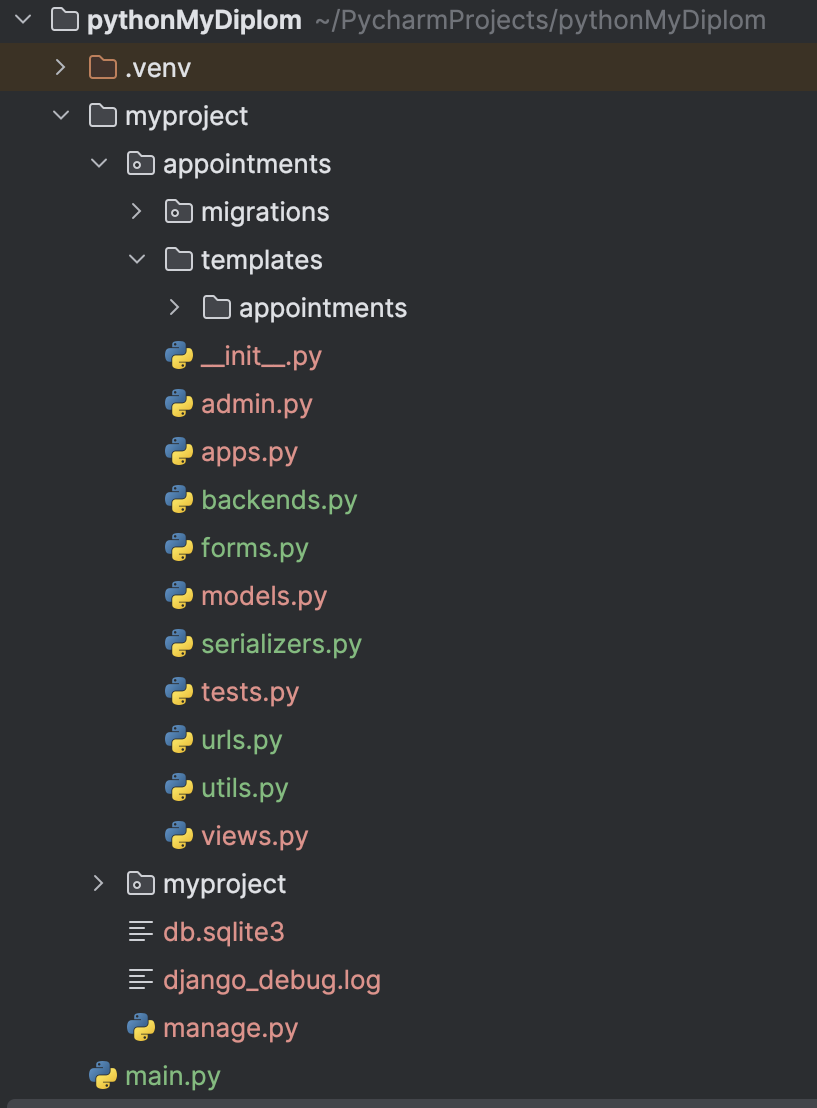
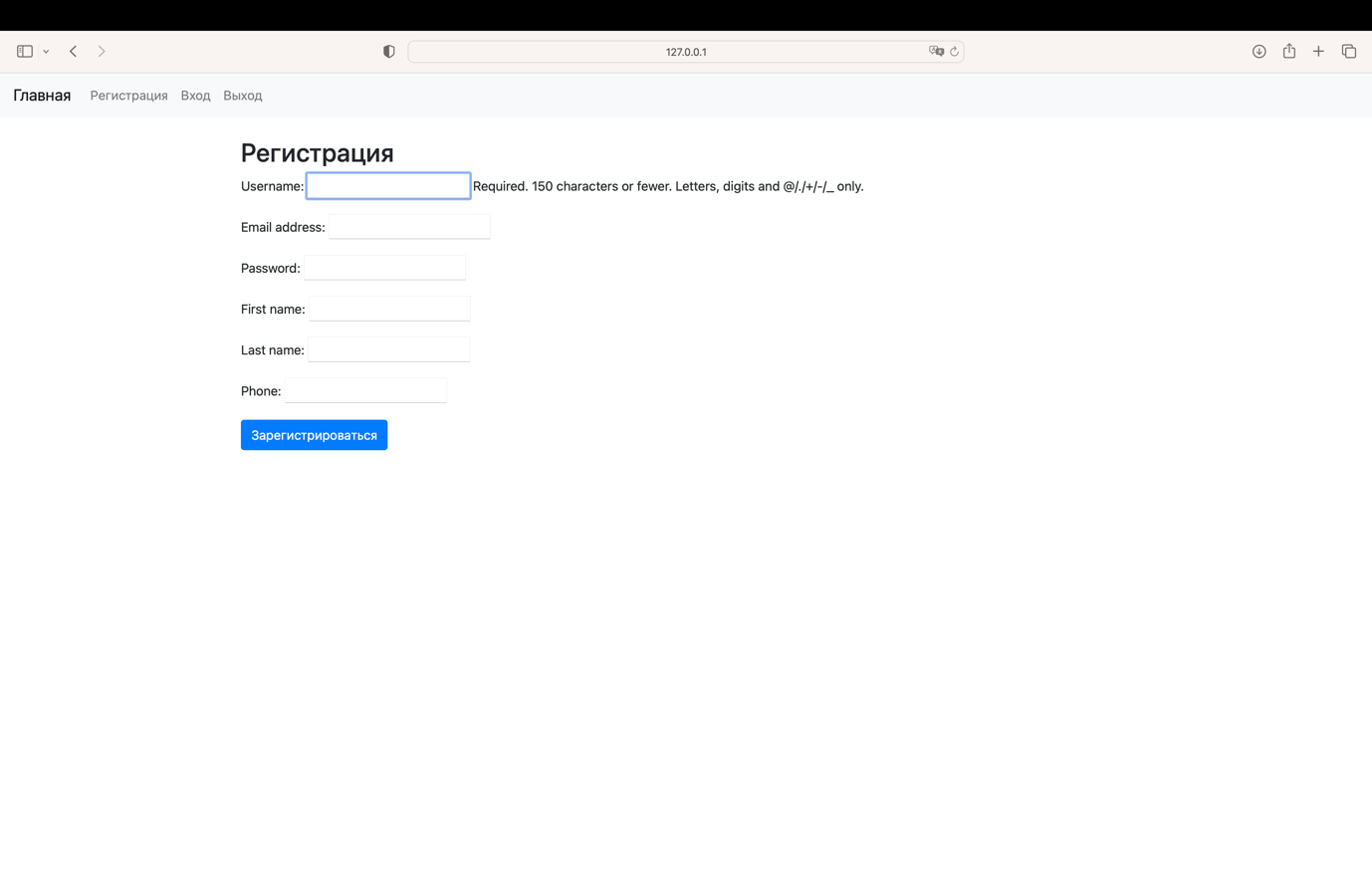


Рис.1 – Структура проекта

Выполнил функционал по регистрации и аутентификации пользователей.

  
  
Рис.2 – Регистрация пользователей

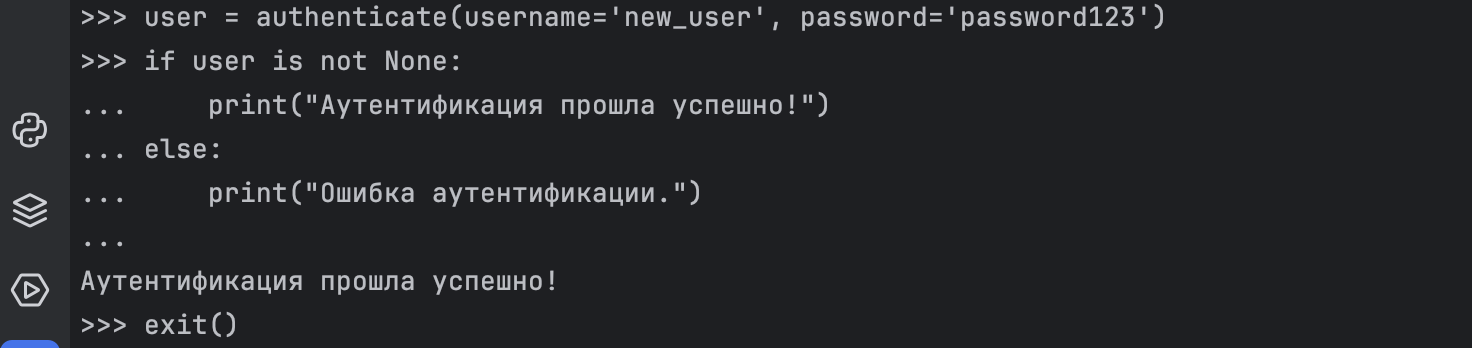


Рис.3 – Проверка аутентификации

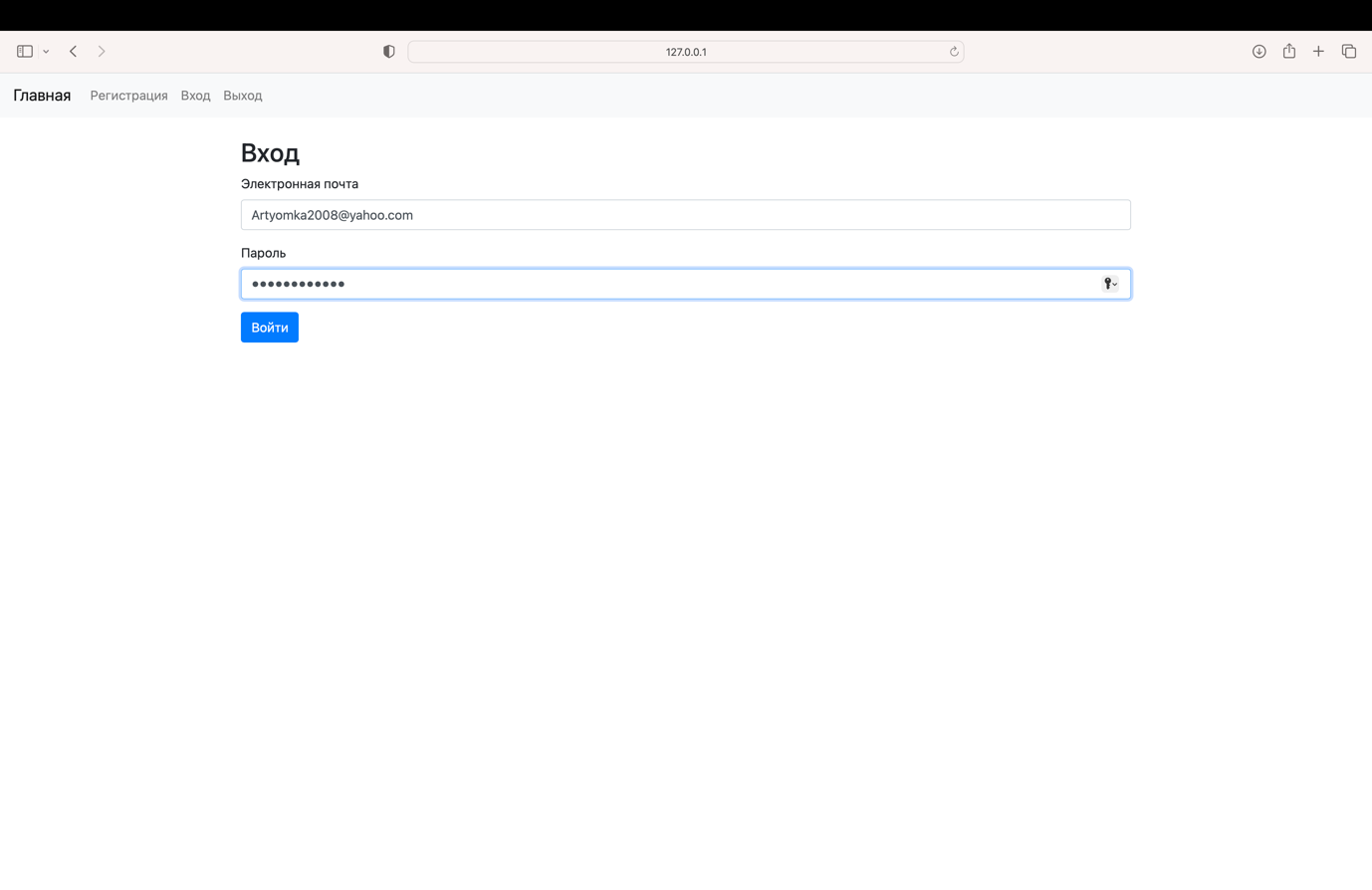


Рис.4 – Вход по email и паролю

Реализован функционал Личного кабинета пользователя с формированием новой заявки

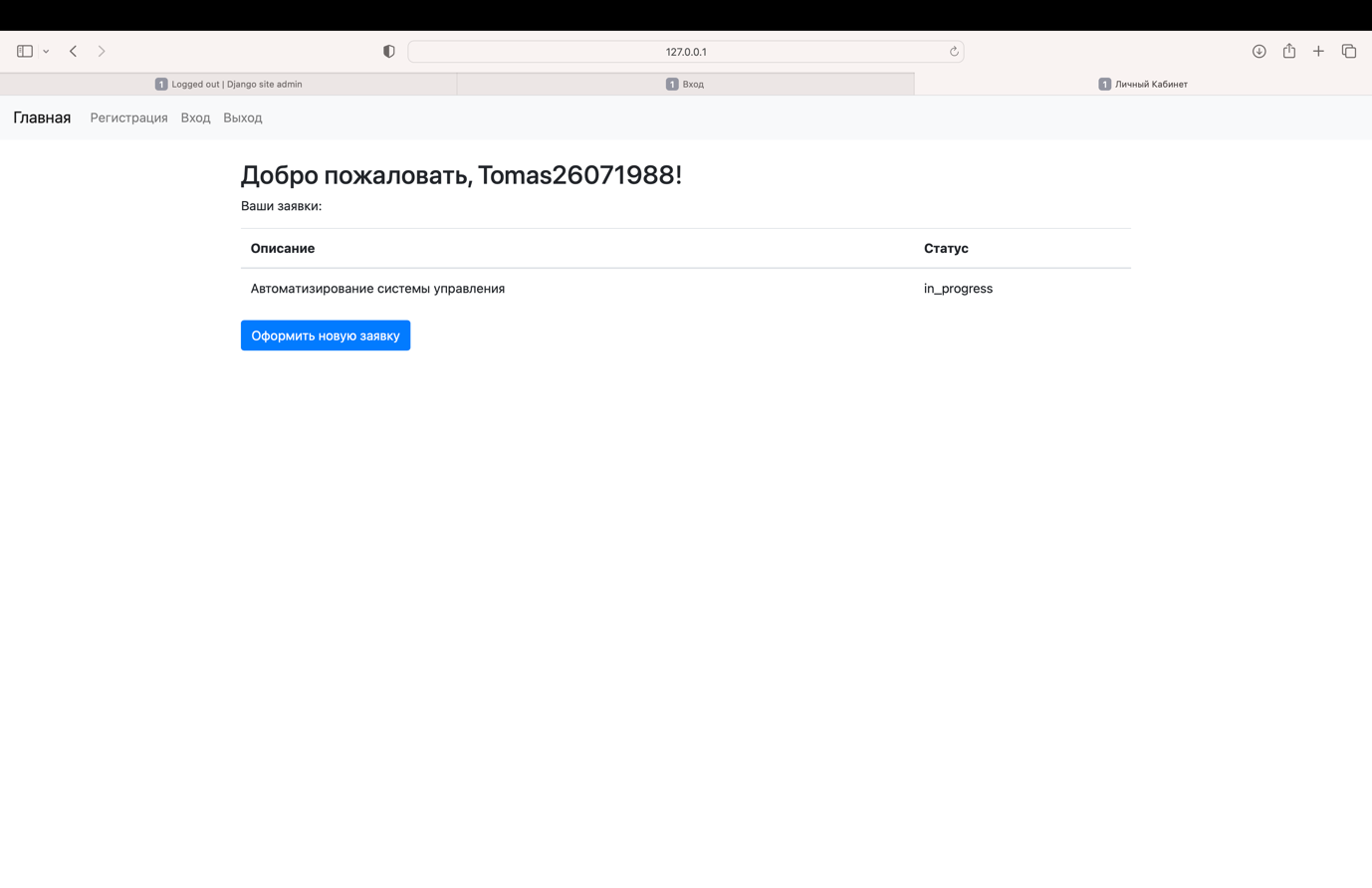


Рис.5 – Личный кабинет

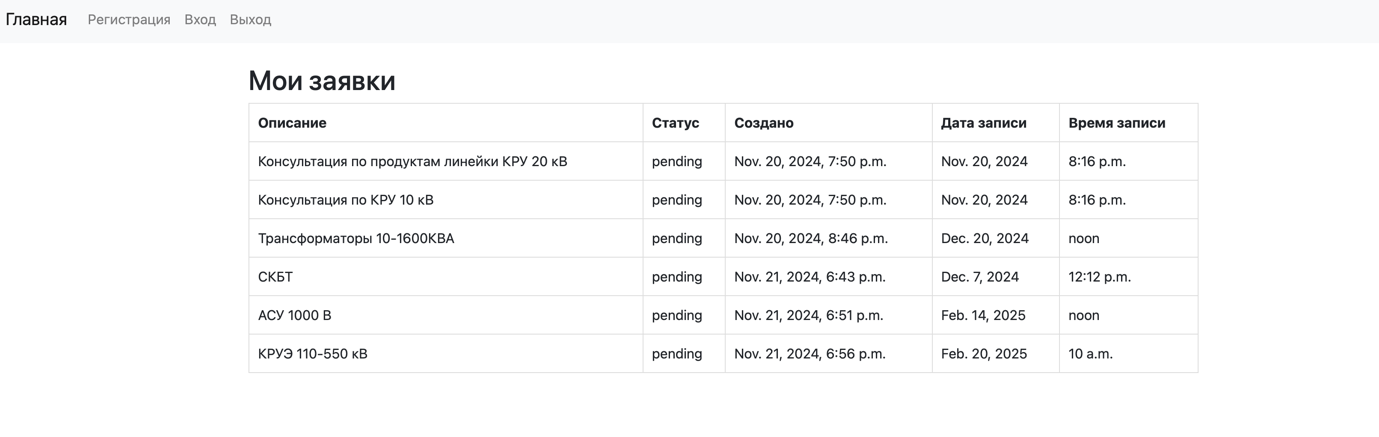
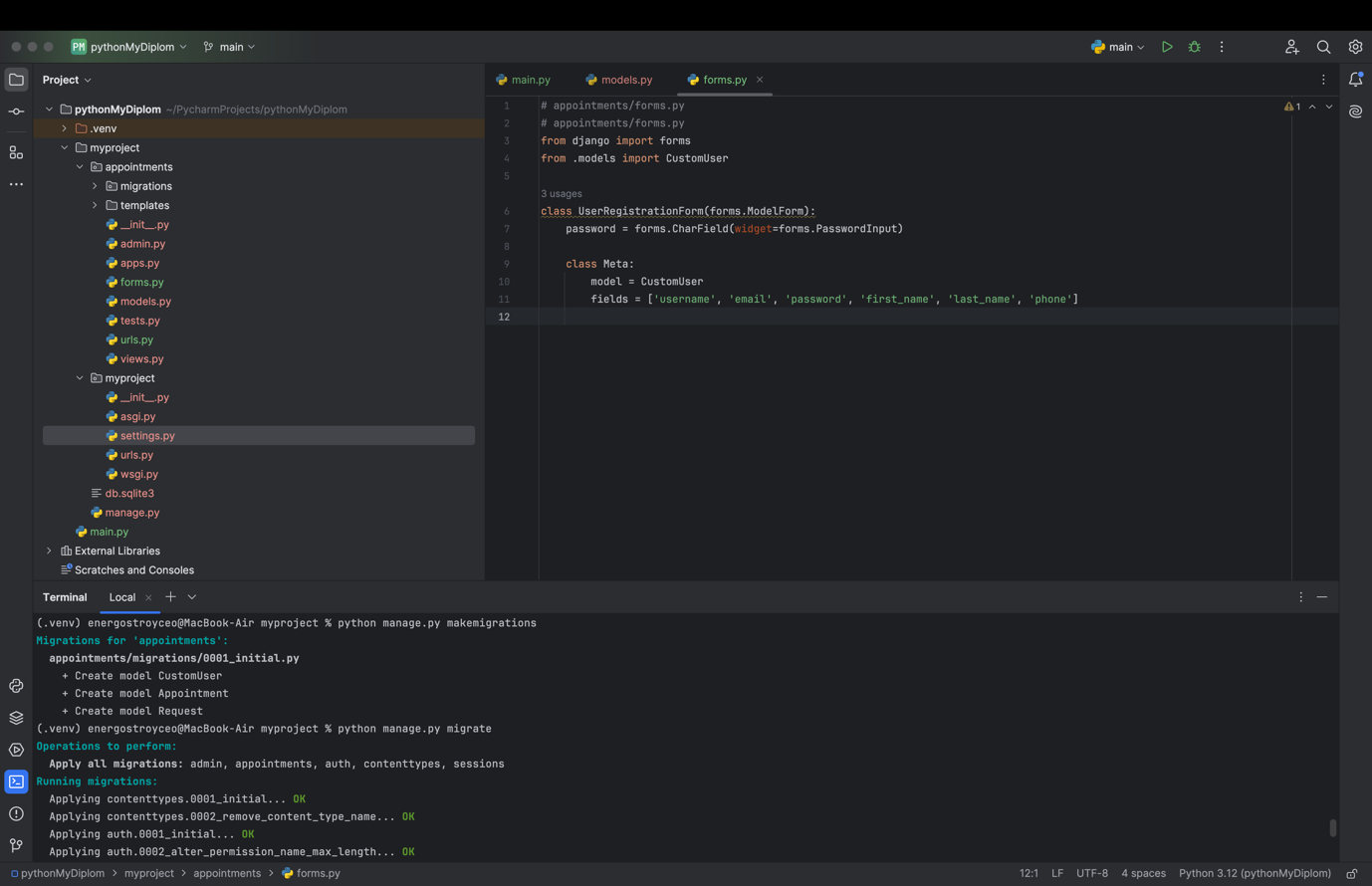


Рис.6 – Портал заявок пользователя

Использована легковесная, встраиваемая реляционная база данных SQLite, которая хранит все данные в одном файле (db.sqlite3). Выполнена настройка базы данных, определена `settings.py`, а также созданы необходимые модели и выполнены миграции:



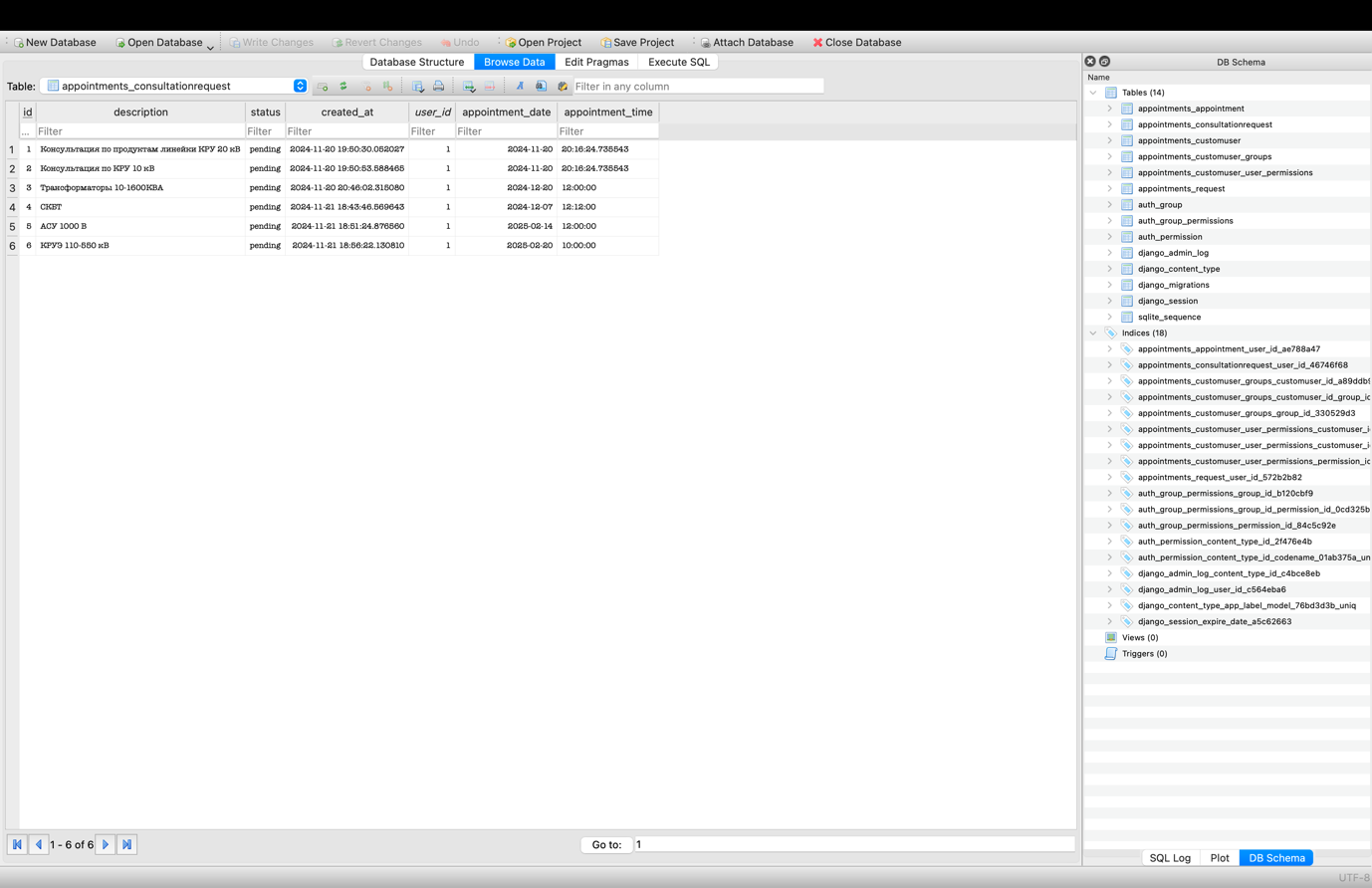


Рис.7 – СУБД SQLite Рис.8 - Миграции

Выполнено и проверено добавление пользователей через **QuerySet** запросы.

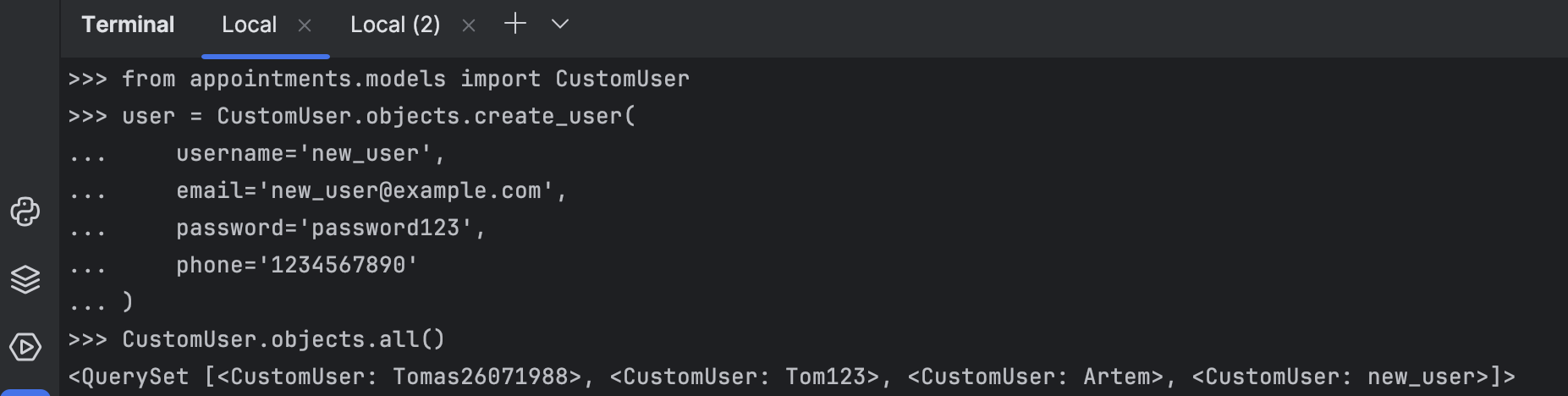


Рис.8 – Добавление пользователя в СУБД через QuerySet

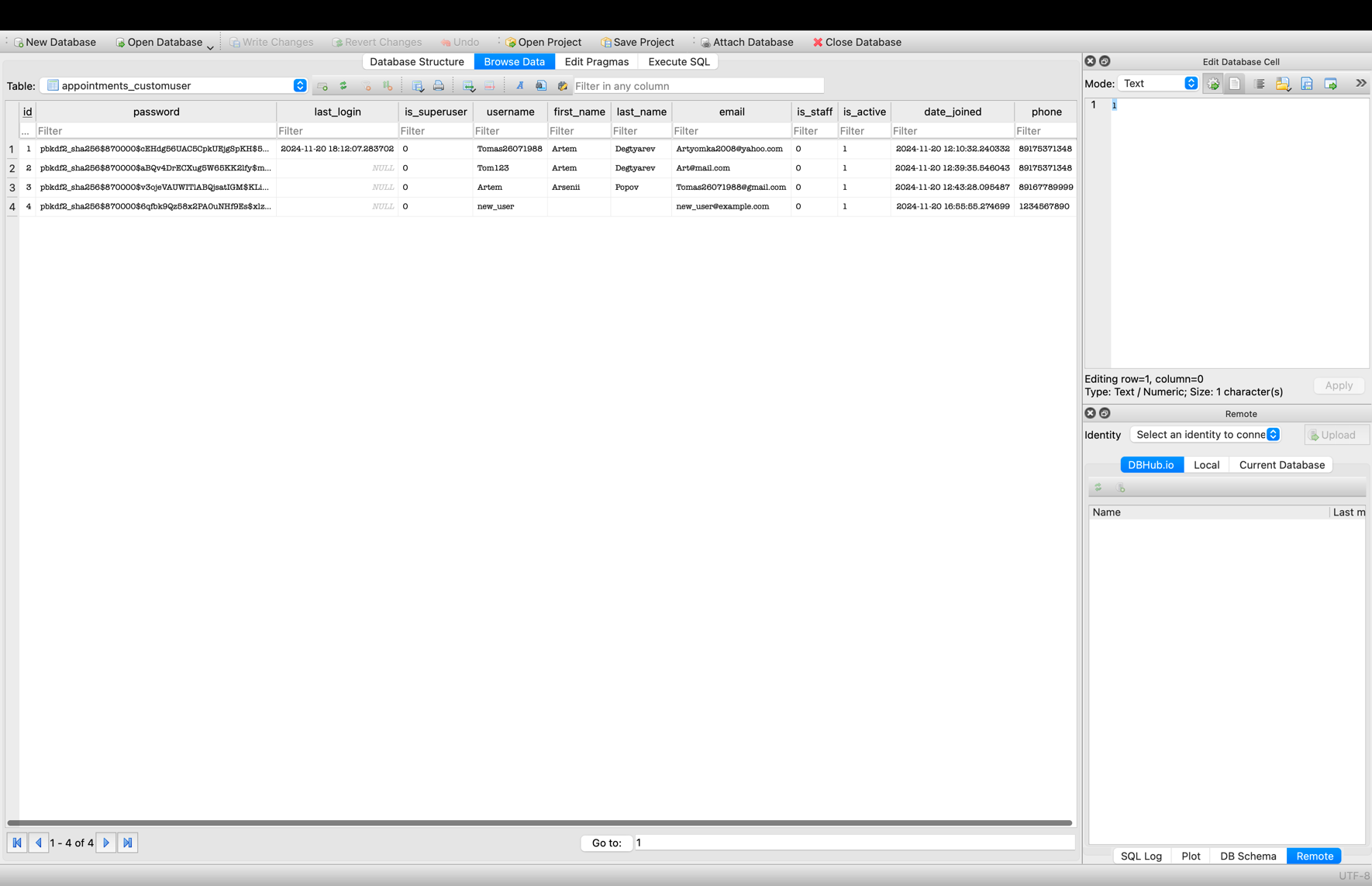


Рис.9 – Новый пользователь отразился под id 4 в БД

Реализован функционал отправки СМС о запланированной консультации в режиме демонстрации.

*Комментарий: для полноценной реализации требуется купить номер на* [*https://www.twilio.com/en-us/phone-numbers*](https://www.twilio.com/en-us/phone-numbers)*. Также можно реализовать на Яндексе, необходимо купить номер и зарегистрироваться. Затем внести изменения в код.*

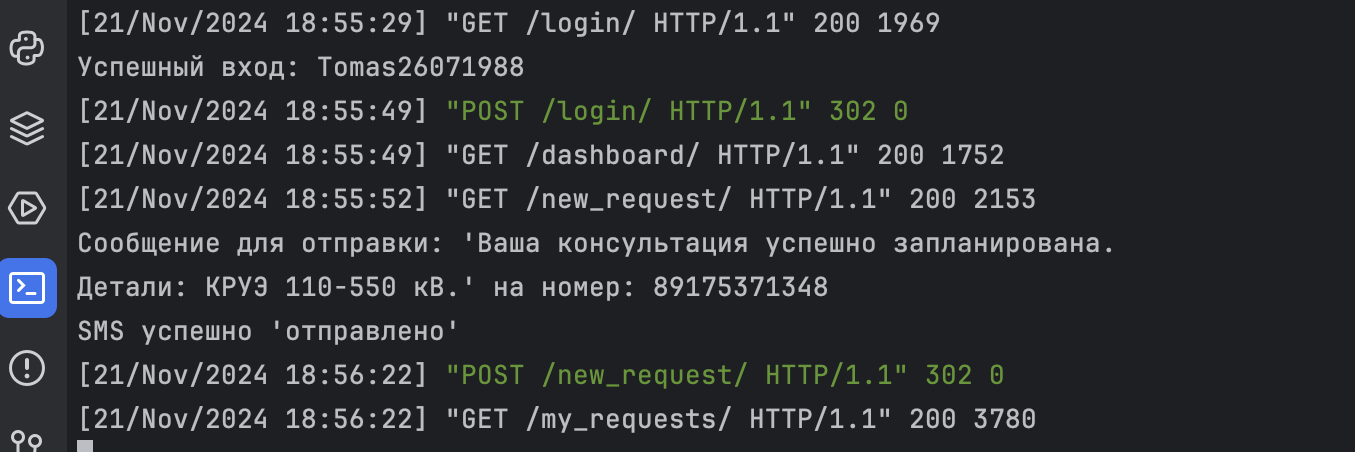
**

Рис.8 – Отправка СМС пользователю

Реализована интеграция с API Django Rest Framework (DRF) с методами HTTP, такие как GET, POST, PUT, PATCH. Что позволяет удобно взаимодействовать пользователю с серверной частью приложения, видеть заявки из личного кабинета пользователя с возможностью их редактирования.

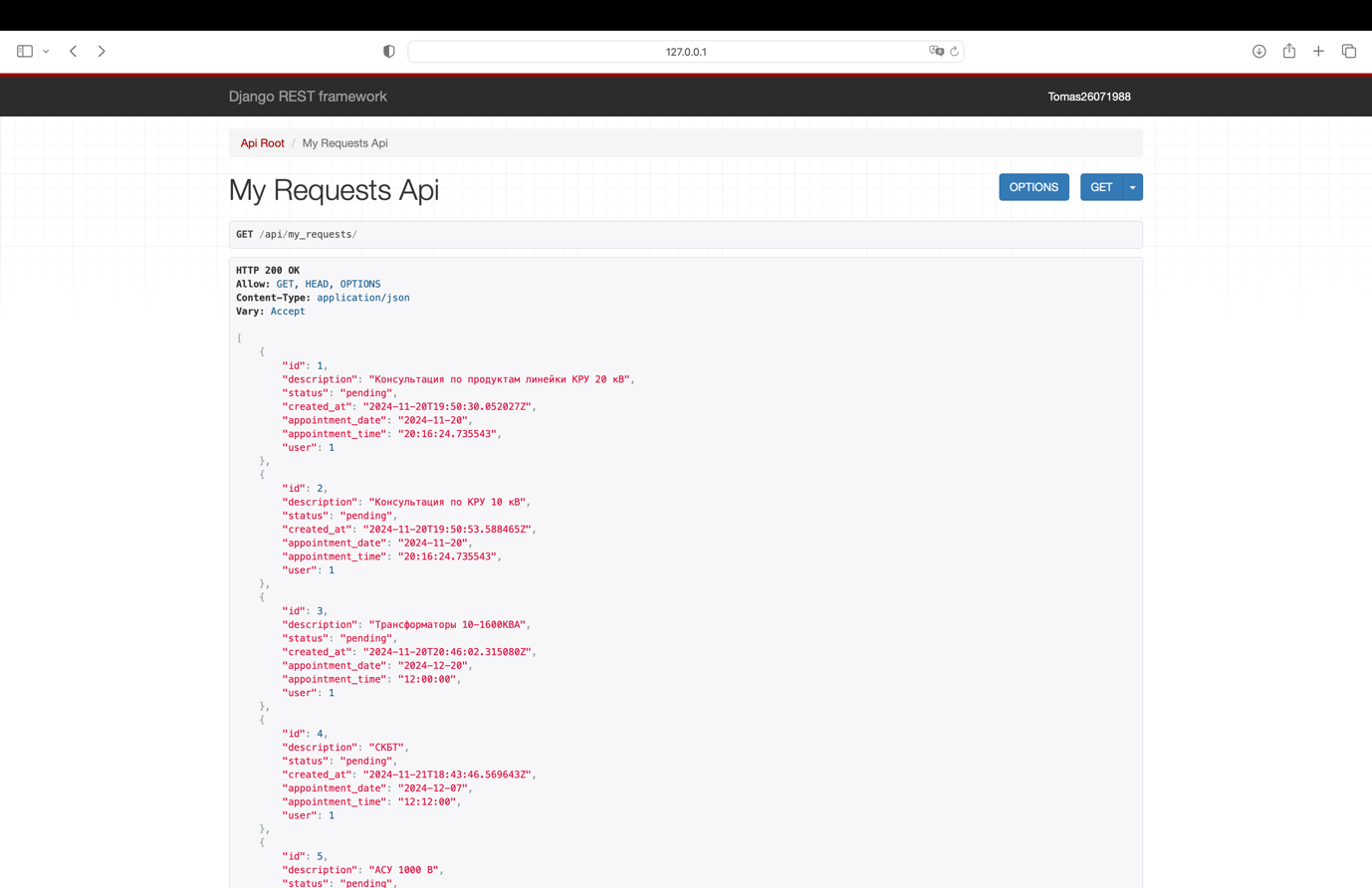


Рис.9 – Интеграция с API DRF

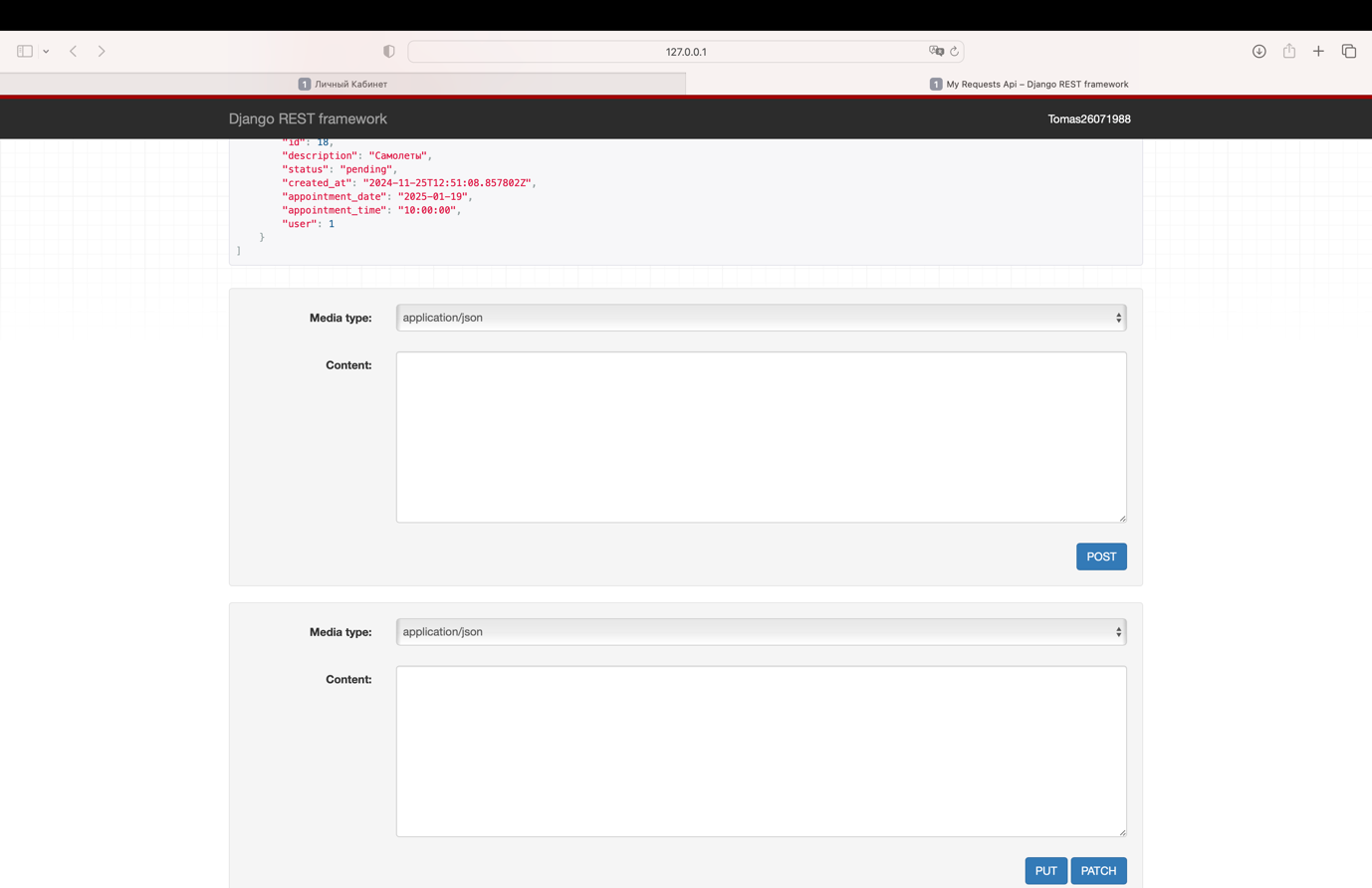


Рис.10 – Пример реализованных методов HTTP

Административная панель: интерфейс для администраторов, позволяющий управлять заявками и пользователями, а также просматривать статистику использования.

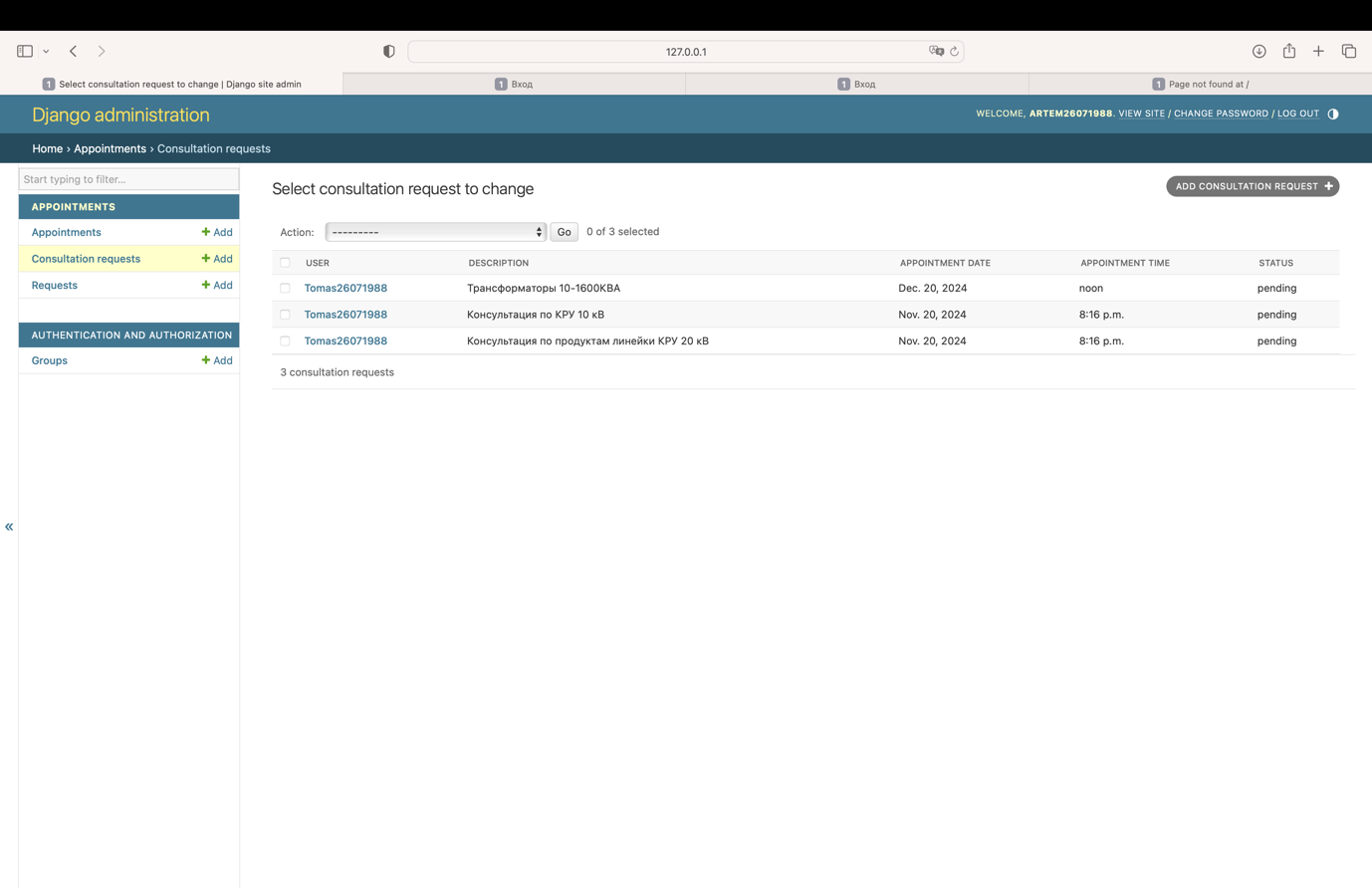


Рис.11 – Админ-панель

Создано логирование для приложения, которая помогает понять, как приложение работает, выявлять и устранять проблемы, а также обеспечивать безопасность. При работе приложения все записи логов осуществляются в файле django\_debug.log.

5.Заключение  
  
Обзор выполненной работы  
  
Проектирование и разработка веб-приложения, которое позволяет пользователям регистрироваться с внесением данных о себе, отправлять заявки на консультации и записываться на встречи в календаре. После записи получать SMS-напоминаний о запланированных встречах.  
Реализованное приложение соответствует требованиям и демонстрирует необходимый функционал.  
  
  
Дальнейшие планы  
  
Будет полезным продолжать расширять и дальше возможности приложения для повышения клиент-ориентированности. После внедрения приложения на сайт моей компании потребуется адаптация и интеграция с сайтом компании и перечнем услуг по консультациям, согласно дизайну сайта, интерфейс и добавить дополнительные функции и ссылки на документацию по бизнес-продуктам. Зарегистрировать и купить номер телефона для внесения в приложения по смс-услугам.

# Приложение 1. Список необходимых библиотек

Python 3.12

PyCharm 2024.1.1

Pip== 23.2.1

Django==5.1.3

djangorestframework==3.15.2

twilio==9.3.7