**РЕФЕРАТ**

Записка с., рис., табл., источников, прил.

АКУСТИКА, ПРОИЗВОДСТВО, ШУМ, ПОЛЬЗОВАТЕЛИ, DJANGO, PYTHON

Объектом разработки является блог с функцией орфографического контроля текста, реализованный с применением фреймворка Django

Цель работы – создание блога с функцией орфографического контроля текста, реализованного с применением фреймворка Django, выполняющего следующие основные функции:

* регистрацию пользователей;
* авторизацию пользователей;
* хранение информации о пользователях;
* создание и хранение проектов;
* создание и хранение измерений;
* изменение прав пользователей на создание проектов.

Также выделяются другие функции:

* формирование диаграммы направленности и ее интерактивная отрисовка;
* формирование графика истории измерений;
* сигнализация об отклонении интенсивности измеренного шума от нормальных значений.

В результате разработки спроектировано, реализовано и протестировано веб-приложение, реализующее вышеперечисленные функции.

Просматривать список проектов могут только зарегистрированные пользователи, создавать проекты – пользователи, имеющие статус проектировщика – отобранные администратором пользователи.

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ 6

ОСНОВНАЯ ЧАСТЬ 7

1. Анализ требований и уточнение спецификаций 7

1.1 Анализ задания и выбор технологии, языка и среды разработки 7

1.2 Разработка диаграммы вариантов использования 8

1.3 Разработка концептуальной модели предметной области 15

1.4 Анализ структуры представления данных 15

1. Проектирование структуры и компонентов программного продукта 17

2.1 Разработка даталогической модели предметной области 17

2.2 Разработка структурной схемы программного продукта 17

2.3 Разработка пользовательского интерфейса 18

2.4 Разработка алгоритма орфографического контроля текста 25

2.5 Разработка диаграммы компоновки программного продукта 27

1. Выбор стратегии тестирования и разработка тестов 28

3.1 Выбор стратегии тестирования 28

3.2 Тестирование методом предположения об ошибке 28

3.3 Оценочное тестирование 33

ЗАКЛЮЧЕНИЕ 37

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 38

ПРИЛОЖЕНИЕ А Техническое задание на разработку 39

ПРИЛОЖЕНИЕ Б Руководство пользователя 46

ПРИЛОЖЕНИЕ В Листинг фрагмента программного продукта 53

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЕ И СОКРАЩЕНИЯ

В настоящей РПЗ применяются следующие определения, обозначения и сокращения:

* CSS – Cascading Style Sheets – формальный язык описания внешнего вида документа
* Django – фреймворк для создания веб-приложений на языке Python
* HTML – HyperText Markup Language – стандартизированный язык разметки документов для просмотра веб-страниц в браузере
* MVC – Model-View-Controller – схема разделения данных приложения и управляющей логики на 3 отдельных компонента: модель, представление и контроллер
* MVT – Model-View-Template – модификация MVC, используемая в Django [1]
* Python – динамический, рефлексивный, интерпретируемый высокоуровневый язык программирования
* БД – база данных
* ООП – Объектно-ориентированное программирование – парадигма программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса [2]
* ТЗ – техническое задание
* ЯП – язык программирования

ВВЕДЕНИЕ

Целью преддипломной практики является изучение студентами в производственных условиях особенностей изготовления программных и программно-аппаратных информационных систем, а также вопросов организации их производства.

Задачей преддипломной практики является формирование компетенций, предусмотренных основной профессиональной образовательной программой на основе ФГОС ВО по направлению подготовки 09.03.03 Прикладная информатика, а также реализация этапов проектирования приложения ВКРБ и описание структуры разрабатываемого продукта.

**Анализ задания и выбор технологии, языка и среды разработки**

Согласно техническому заданию, необходимо разработать веб-приложение для анализа шума на производстве с использованием фреймворка Django. Этот фреймворк позволяет быстро вести разработку, поскольку предлагает к использованию целый спектр готовых решений (реализация концепции ORM, наличие API доступа к БД), повышает степень повторного использования кода. При использовании данного фреймворка результатом работы будет веб-приложение, клиентская часть которого будет запускаться в качестве страницы в браузере, а серверная часть будет работать под управлением операционной системы Linux. Преимущества Linux: это бесплатно, Linux потребляет гораздо меньше аппаратных ресурсов (в сравнении с Windows) [4], хорошо настроенный Linux практически не нуждается во вмешательстве системного администратора. Ниже описаны принятые перед началом разработки решения.

Предопределенный в техническом задании фреймворк Django построен на объектно-ориентированном подходе, из чего вытекает соответствующий подход к разработке. Парадигма ООП позволяет обеспечить максимальную эффективность разработки, так как предоставляет наиболее естественную декомпозицию предметной области [2].

В качестве модели разработки выбрана спиральная схема разработки, поскольку она наиболее простым образом позволяет масштабировать проект. Выбор этой модели позволяет:

1) сократить время до появления первых версий программного продукта;

2) ускорить формирование и уточнение спецификаций за счет появления практики использования продукта;

3) вносить изменения в код при появлении новых требований на этапе каждой итерации.

Языком программирования выбран язык Python версии 3.9.7, используемый в работе фреймворка Django. Данный ЯП является динамическим интерпретируемым высокоуровневым языком с поддержкой ООП, что позволяет использовать объектно-ориентированный подход программирования в ходе разработки. Он широко используется в веб-разработке, так как он обеспечивает высокую скорость разработки, а также предоставляет большое количество библиотек и модулей, реализующих большинство необходимых функций, которые можно сразу использовать в разработке программного продукта.

В качестве архитектурного шаблона проектирования выбран шаблон MVT, непосредственно реализуемый выбранным фреймворком Django версии 3.2.9. Данный шаблон предоставляет многоуровневую архитектуру, которая разделяет приложение на уровни представления, логики и информации, таким образом, что модификация каждого компонента может осуществляться независимо, что значительно упрощает разработку и повышает возможность повторного использования кода [1].

В качестве БД при разработке приложения выбрана SQLite3, используемая по умолчанию при создании нового проекта в установленном техническом задании фреймворке Django. Данный выбор обусловлен тем, что данная БД обладает необходимым функциональность для разработки и тестирования созданного веб-приложения, так как обеспечивает высокую скорость простых операций выборки данных, высокую экономичность использования ресурсов, а также безопасность сохраненных данных [5]. Однако при изменении требований во время разработки программного продукта, фреймворк Django предоставляет возможность изменить используемую БД с SQLite3 на ее более многофункциональные, но более медленные аналоги, например на БД MySQL или PostgresSQL.

В качестве языков для реализации клиентской части выбраны язык разметки HTML и язык таблицы стилей CSS. Данный выбор обусловлен тем, что альтернатив этим языкам на данный момент не существует [6].

В качестве среды разработки выбран текстовый редактор PyCharm CE версии 21.2.2. Данная среда разработки является оптимальной для разработки на Django и Python [7] из-за ее простоты и гибкости, множества настроек, наличия встроенного терминала, возможности пошаговой отладки и большого выбора подключаемых плагинов, а именно подсвечивание синтаксиса, автоматическое подключение библиотек, и автоматический анализатор кода.

**1.3 Разработка концептуальной модели предметной области**

В разрабатываемой веб реализации блога цель основного варианта использования заключается в проведении пользователями измерений в рамках проектов и ведении мониторинга.

Соответственно можно выделить 3 основных понятия:

⎯ пользователь;

⎯ измерение;

⎯ проект.

Каждое измерение принадлежит пользователю и проекту, проект – пользователю. На основании проведенного анализа составлена концептуальная модель предметной области, представленная на рисунке 2

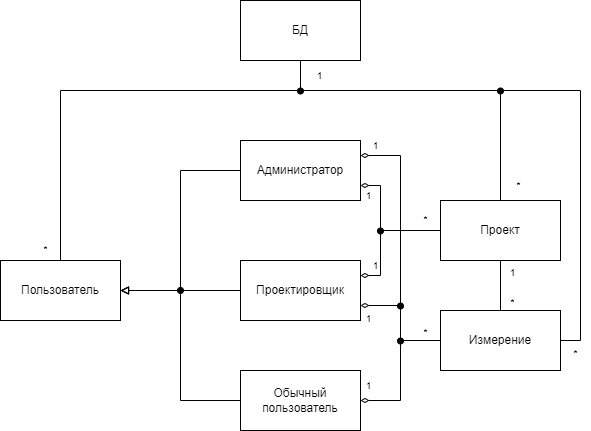


Рисунок 2 - Концептуальная модель предметной области

**1.4 Анализ структуры представления данных**

Произведя анализ концептуальной модели предметной области и техническое задание, установлено, что для реализации основного функционала веб-приложения необходимо хранить данные о пользователях, проектах и измерениях.

Выделены следующие сущности:

- Проекты: смысловое объединение измерение, в рамках которого ведется мониторинг показателей;

- Измерение: для хранения аудиоданных, пользователя, который провел эксперимент, времени измерения.

- Пользователь: для авторизации в системе и создании проектов и измерений.

Остальные понятия, полученные при анализе предметной области, не являются сущностями, из-за чего их следует причислить к сервисам. Выделенные в результате анализа сущности и их атрибуты представлены в таблице 2.

Таблица 2 – Таблица сущностей и атрибутов

|  |  |  |
| --- | --- | --- |
| **Наименование таблицы (сущности)** | **Атрибут** | **Информация об атрибуте** |
| Проект | ID | Первичный ключ |
| Название | Заголовок проекта (строка) |
| Текст | Описание проекта (текстовое поле) |
| Дата создания | Дата и время создания проекта |
| Дата изменение | Дата и время изменение проекта |
| ID пользователя | Внешний ключ, указывающий на пользователя, который создал проект |
| Измерение | ID | Первичный ключ |
| Название | Название измерения (строка) |
| Дата | Дата и время измерения |
| ID пользователя | Внешний ключ, указывающий на пользователя, который провел измерение |
| ID проекта | Внешний ключ, указывающий на проект, к которому относится измерение |
| Пользователь | ID | Первичный ключ |
| Логин | Имя пользователя (строка |
| Пароль | Пароль, хранящийся в зашифрованном виде (строка) |
| Почта | Почта пользователя (строка типа email) |
| Проектироващик | Логическое поле (истина – пользователь проектировщик, иначе – нет) |

**2.5 Разработка диаграммы компоновки программного продукта**

В связи с тем, что разработка веб-приложения проведена с использованием фреймворка Django, который предполагает архитектурный шаблон MVT, то архитектура приложения заранее предполагает определенную иерархию компоновки. Модель является связующим звеном между базой данных и бизнес-логикой приложения, которую реализуют представления. Формы интерфейса отображают шаблоны. На рисунке 19 представлена разработанная диаграмма компоновки программного продукта.

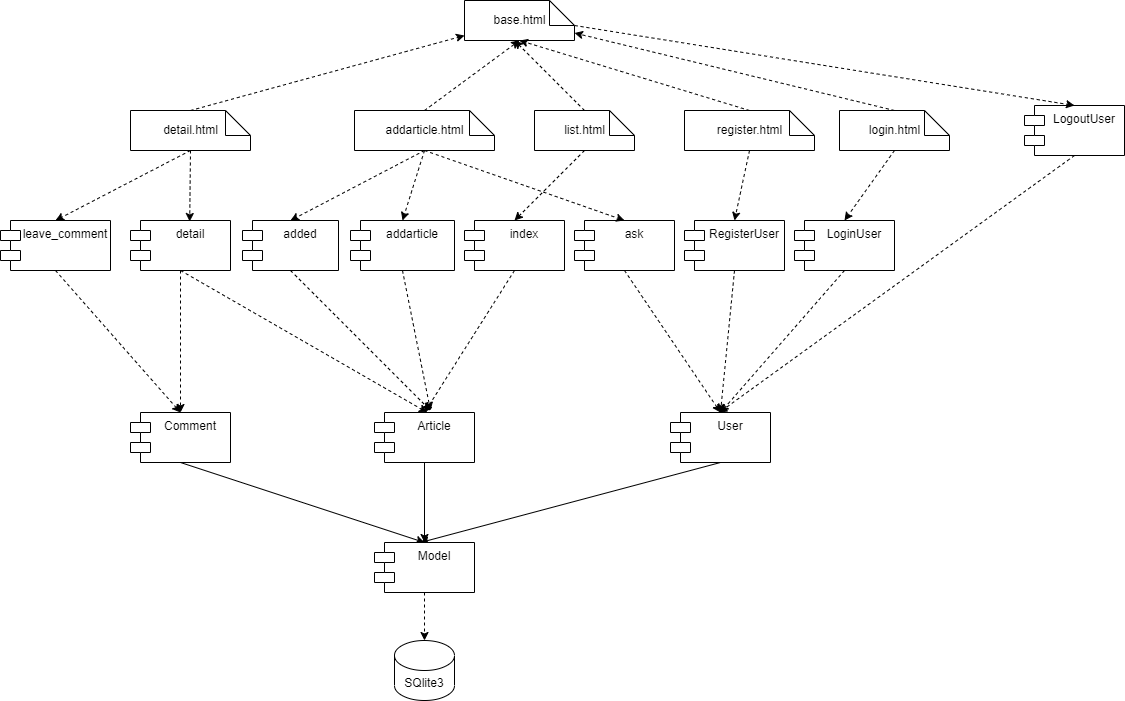


Рисунок 18 - Диаграмма компоновки

**3 Разработка структурной схемы веб-приложения**

Структуру разрабатываемого программного обеспечения можно представить в виде серверной части, которая содержит модуль контроллера, реализующего основную логику веб-приложения, и клиентской части, в качестве которой выступают веб-браузер пользователя и модуль представления. Разработанная структурная схема веб-приложения представлена на рисунке 3.

Фреймворк Django написан на языке программирования Python, поэтому его структура соответствует особенностям языка. Создатели реализовали в Django паттерн MVC (архитектура MVC позволяет разработчику работать с визуальным представлением и бизнес-логикой приложения отдельно). При работе с Django более корректно использовать термин MVT — Model-View-Template или модель-представление-шаблон.

Модель – источник информации о данных, в которых содержатся ключевые поля и поведение данных. Обычно одна модель указывает на одну таблицу в базе данных. Django поддерживает базы данных PostgreSQL, MySQL, SQLite и Oracle. Модели содержат информацию о данных. Эти данные представлены атрибутами или полями. Поскольку модель представляет собой простой класс, она ничего не знает о других уровнях Django. Взаимодействие между уровнями происходит через API. Модель отвечает за бизнес-логику, методы, свойства и другие элементы, связанные с манипуляцией данными. Также модели позволяют разработчикам создавать, читать, обновлять и удалять объекты в базе данных.

Представление (view) решает три задачи: принимает HTTP-запросы, реализует бизнес-логику, определённую методами и свойствами, отправляет HTTP-ответ в ответ на запросы. То есть представление получает данные от модели и предоставляет шаблонам (templates) доступ к этим данным или предварительно обрабатывает данные и затем предоставляет к ним доступ шаблонам.

В Django реализован мощный движок шаблонов и собственный язык разметки. Шаблоны представляют собой файлы с HTML-кодом, с помощью которого отображаются данные. Содержимое файлов может быть статическим или динамическим. Шаблоны не содержат бизнес-логики. Поэтому они только отображают данные.

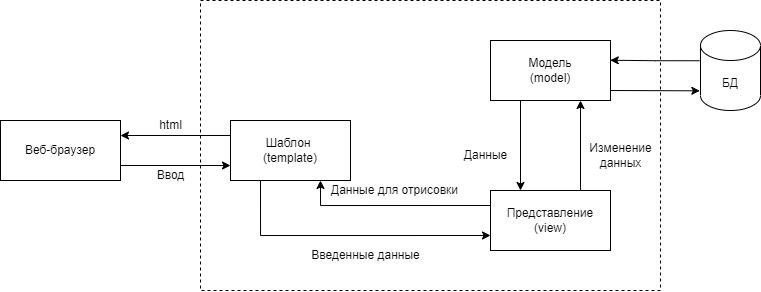


Рисунок 3 - Структурная схема веб-приложения

Веб-браузер реагирует на пользовательские действия, которые главным образом нацелены на получение диаграммы направленности на основе записанного аудиофайла, а также на мониторинг отклонений интенсивности от нормальных значений.

4 Разработка диаграмм классов

5 Разработка форм интерфейса

С учетом анализа предметной области и разработанных диаграмм решено использовать объектно-ориентированный веб-интерфейс.

Интерфейс разработанного веб-приложения выполнен при помощи средств языка разметки HTML, языка стилей, а также встроенного в Django шаблонизатора, который позволяет представлять формы интерфейса в качестве объектов языка Python, которые интерпретируют написанный внутри код

На рисунке 5 представлен граф состояний интерфейса

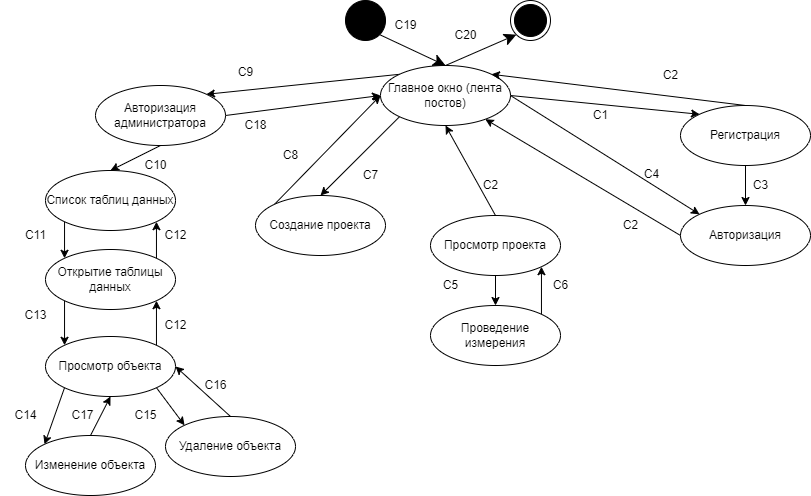


Рисунок 5 - Граф состояний интерфейса

Условные обозначения, принятые на рисунке 5:

* С1 – нажатие кнопки регистрации;
* С2 – нажатие кнопки «проекты» или лого;
* С3 – нажатие кнопки зарегистрироваться;
* С4 – нажатие кнопки авторизации;
* С5 – переход по ссылке добавления измерения;
* С6 – нажатие добавления измерения;
* С7 – переход по ссылке добавления проекта в панели навигации;
* С8 – нажатие кнопки создания поста;
* С9 – посещение адреса root/admin;
* С10 – ввод данных администратора;
* С11 – нажатие названия таблицы данных;
* С12 – нажатие кнопки «назад»;
* С13 – нажатие строчки объекта;
* С14 – ввод новых данных и нажатие кнопки «изменить»;
* С15 – нажатие кнопки «удалить»;
* С16 – нажатие кнопки отмены;
* С17 – нажатие кнопки отмены;
* С18 – посещение главной страницы сайта;
* С19 – вход на сайт;
* С20 – выход с сайта.

Ниже представлены скриншоты разработанных необходимых форм интерфейса

На рисунке 6 представлен список проектов (главная страница)

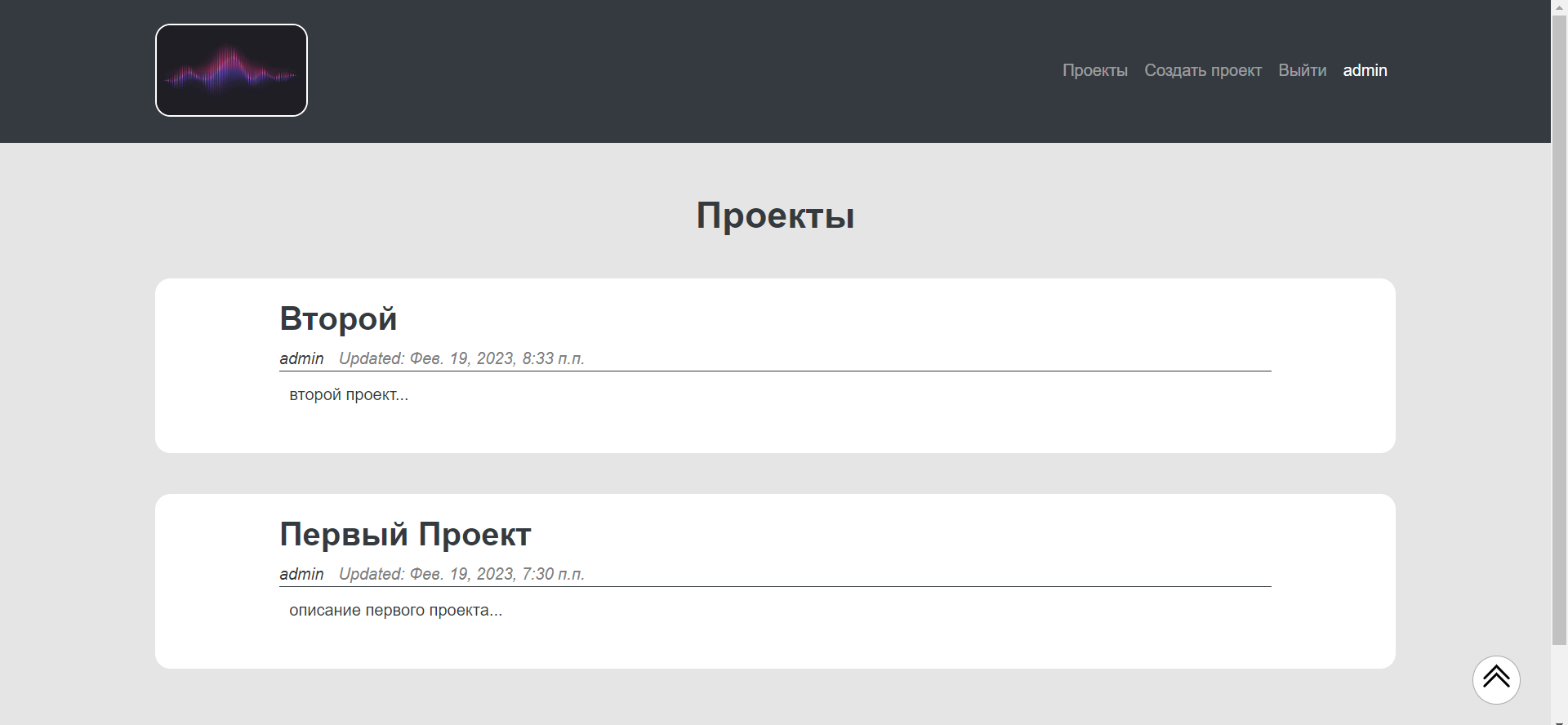


Рисунок 6 – Список проектов

На рисунке 7 представлена форма для регистрации. Пользователю предлагается придумать себе уникальный логин и ввести свой собственный пароль, необходимый для авторизации.

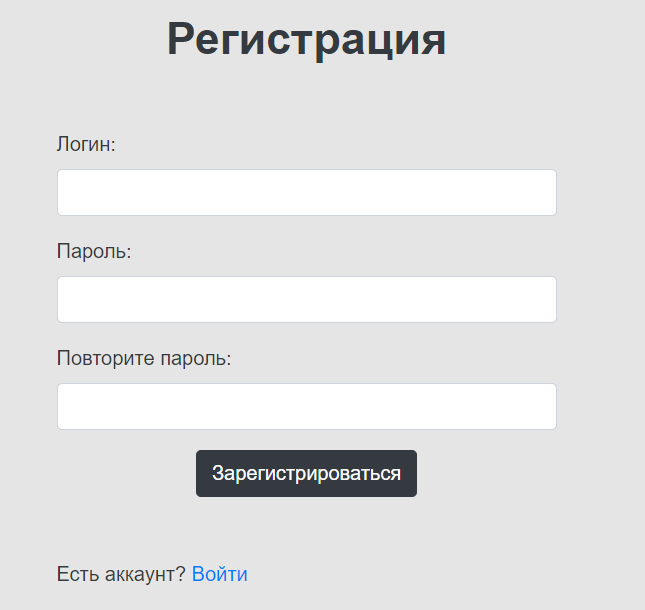


Рисунок 7 - Форма регистрации

На рисунке 8 представлена форма для авторизации, где пользователю необходимо ввести свой логин и придуманный им при регистрации пароль для авторизации в системе.

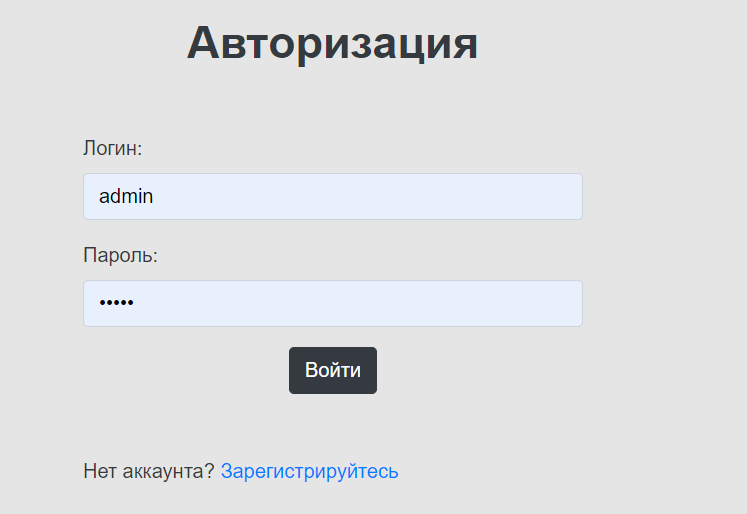


Рисунок 8 - Форма авторизации

На рисунке 9-10 представлена страница просмотра проекта.

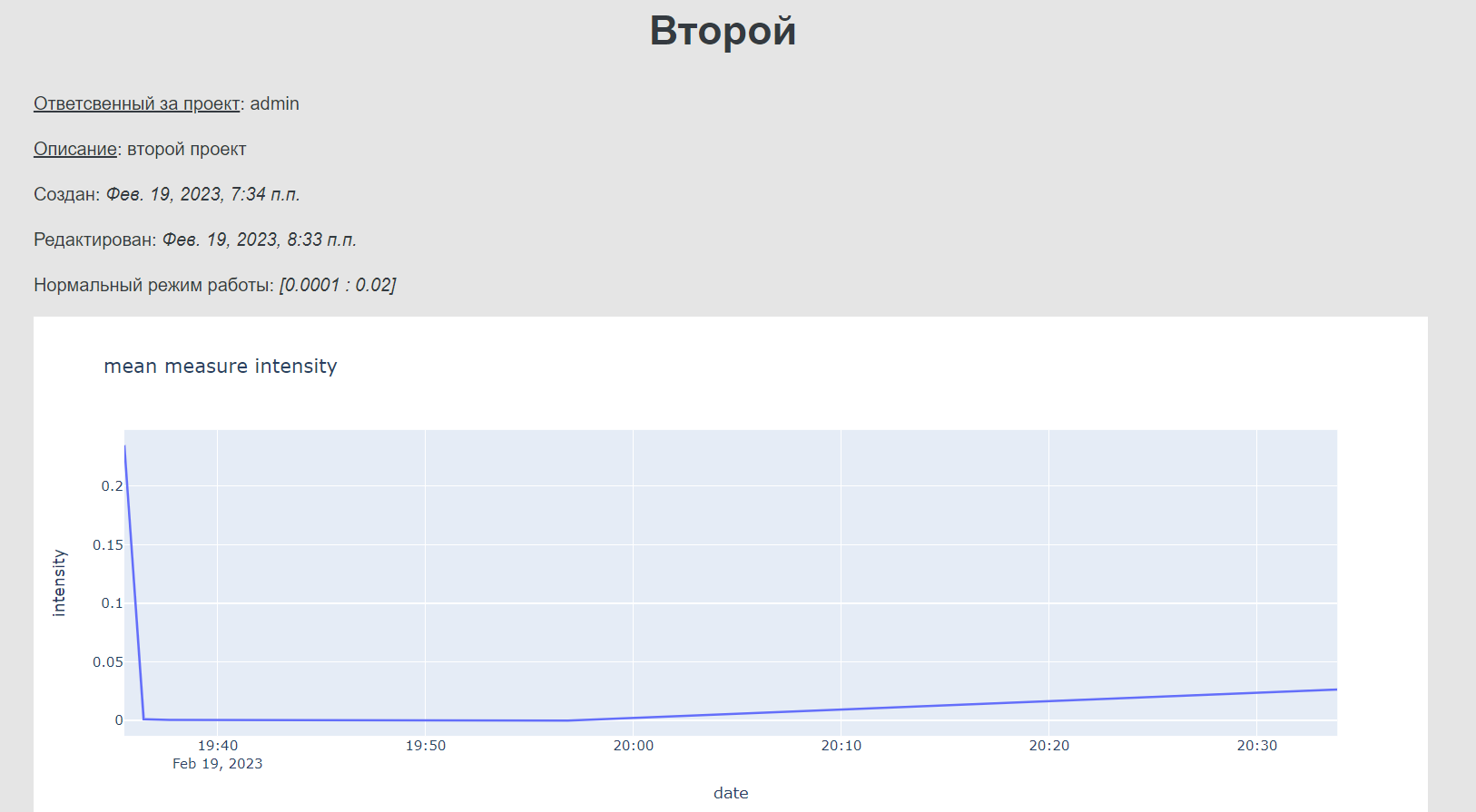


Рисунок 9 - Страница просмотра проекта

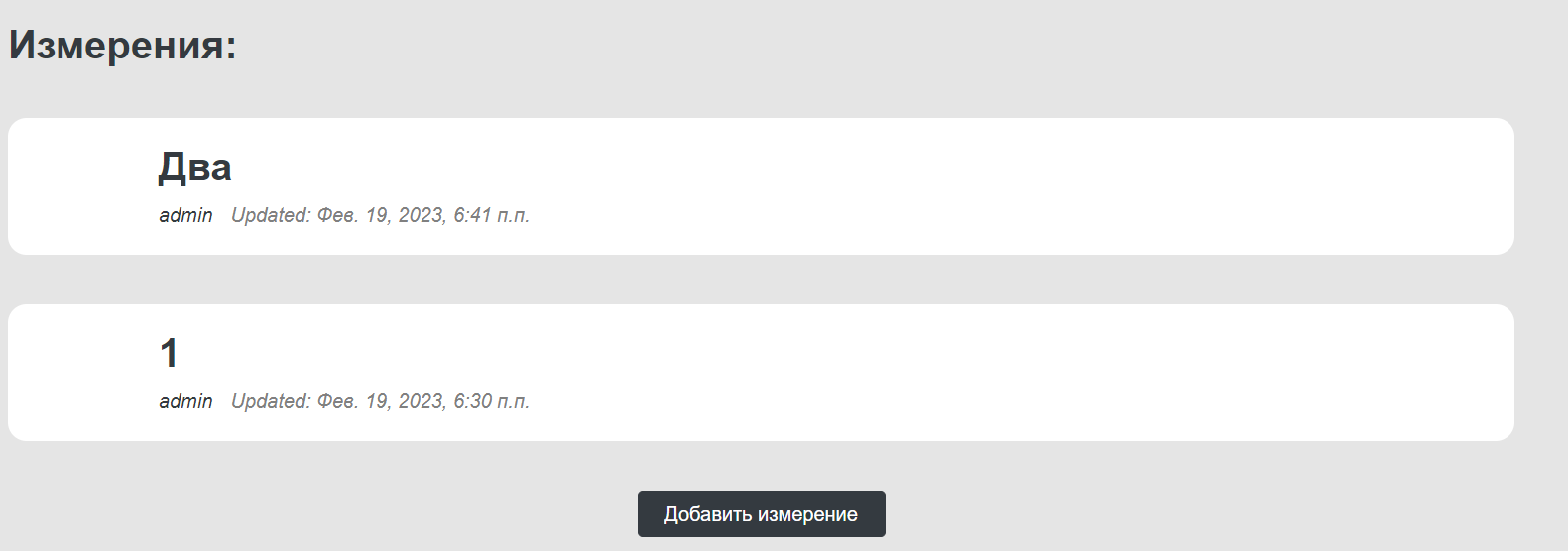


Рисунок 10 - Страница просмотра проекта

На рисунке 10 представлена форма проведения измерений

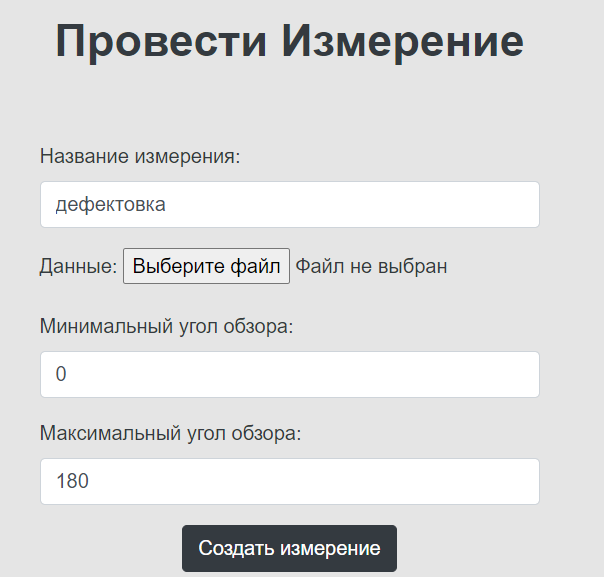


Рисунок 10 - Форма проведения измерения

На рисунке 11 представлена форма создания проекта

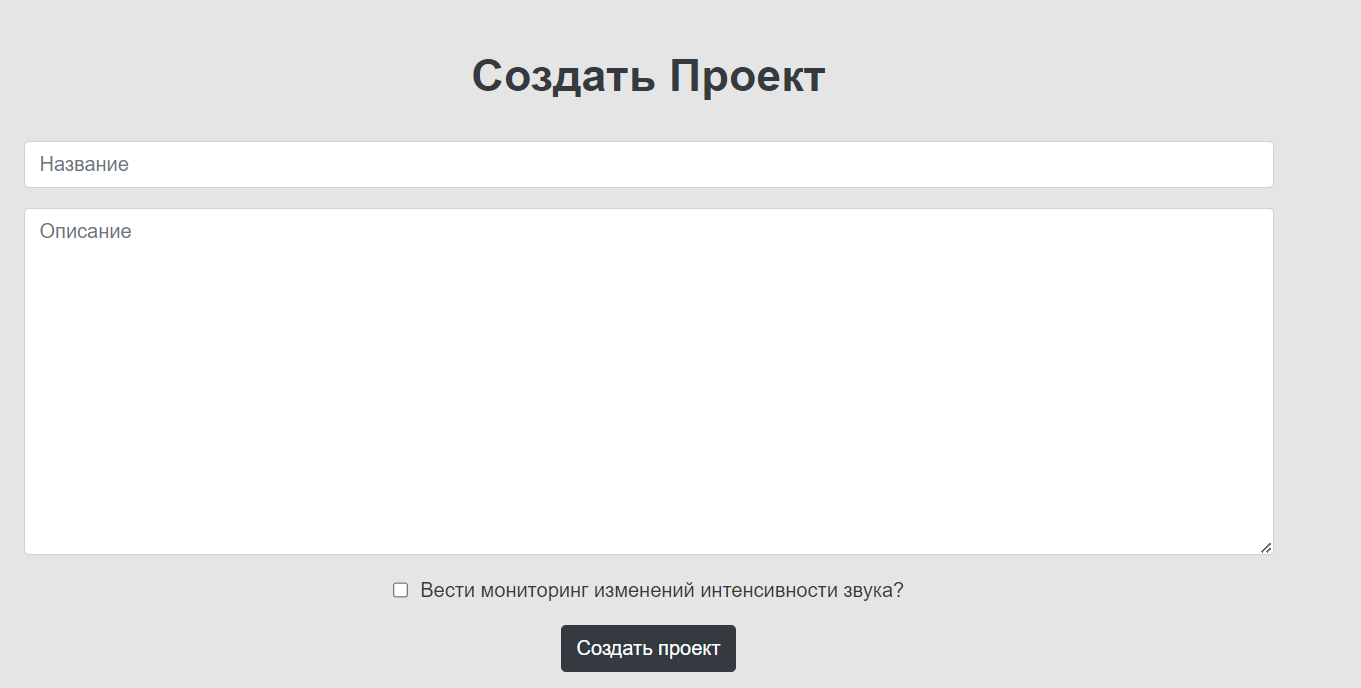


Рисунок 11 - Форма создания проекта

На рисунке 12 представлена форма авторизации администратора

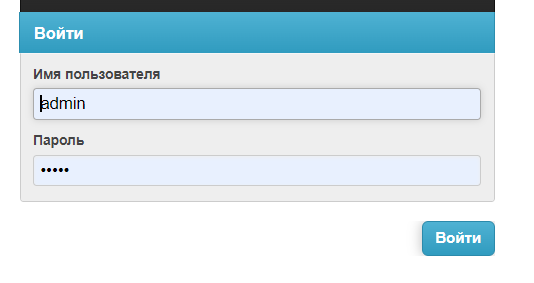


Рисунок 12 - Форма авторизации администратора

На рисунке 13 представлен список таблиц данных

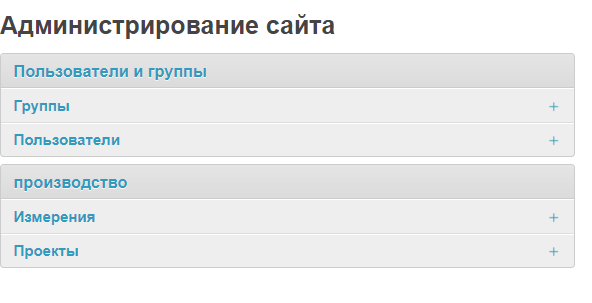


Рисунок 13 - Список таблиц данных

На рисунке 14 представлена таблица проектов

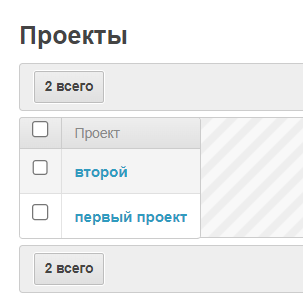


Рисунок 14 - Таблица проектов

На рисунке 15 представлен процесс наделения пользователя статусом проектировщика

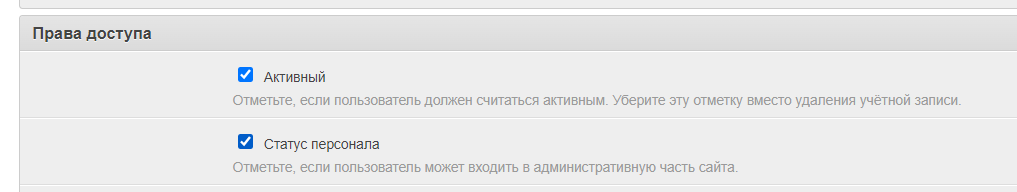


Рисунок 15 - Наделение пользователя статусом проектировщика

На рисунке 16 представлена форма изменения проекта

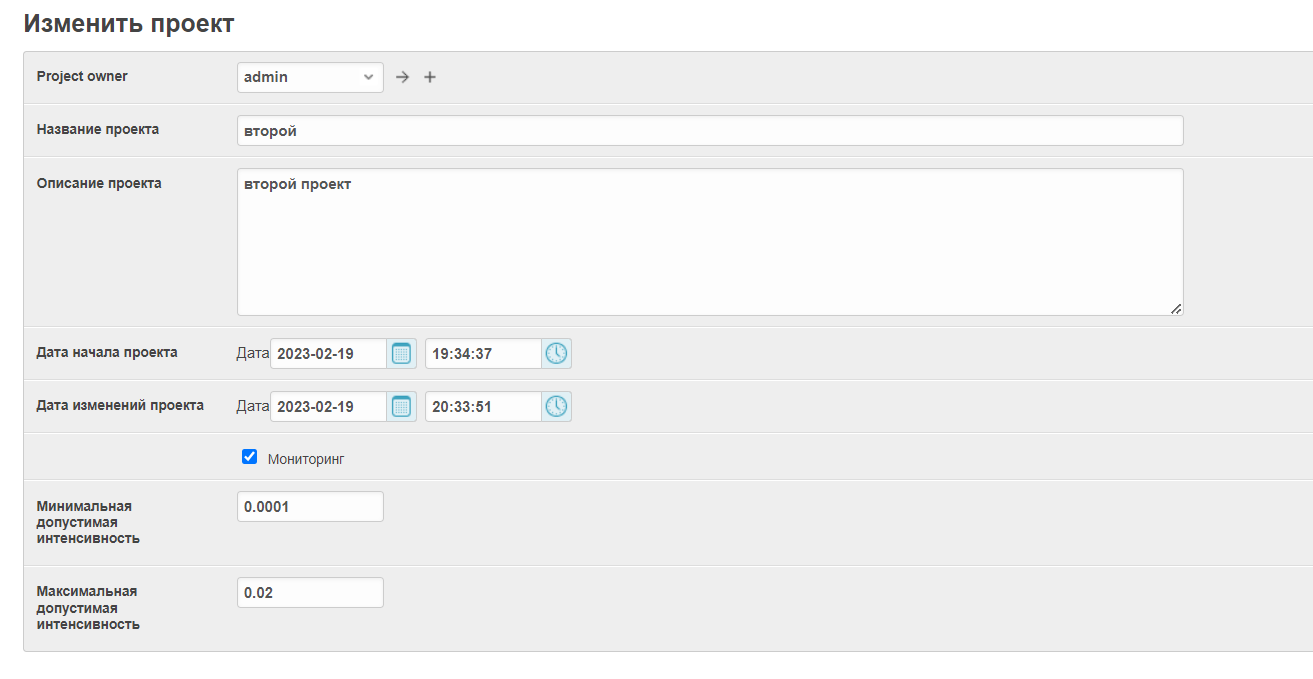


Рисунок 16 - Изменение проекта

**ЗАКЛЮЧЕНИЕ**

В ходе преддипломной практики разработано веб-приложение для анализа шума на производстве, которое позволяет вести плановый мониторинг интенсивности шума, а также формировать диаграмму направленности звука, для выявления его источников.

Заявленная функциональность веб-приложения реализована в соответствии с требованиями технического задания, а в ходе разработки приобретены практические навыки понимать и уточнять спецификации, опыт проектирования и разработки программного продукта клиент-серверной архитектуры среднего уровня сложности, навыки программирования с использованием фреймворка Django, языка Python, языка SQL в SQLite3.

На данный момент веб-приложение реализует основные функции, требуемые для анализа шума на производстве, однако еще не способно вести обработку потоковой информации в режиме реального времени, что оставляет пространство для дальнейшего развития получившегося программного продукта.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Документация фреймворка Django [Электронный ресурс]. – URL: https://docs.djangoproject.com/en/3.2/ (дата обращения 8.11.2021)
2. Иванова Г.С., Ничушкина Т.Н., Пугачев Е.К. Объектно-ориентированное программирование: учебник для вузов. – М.: Изд-во МГТУ им. Баумана, 2001.
3. Тернявская Е. И., Карпова Е. Г. Психология толпы в социальных сетях / Социальный инженер - Москва, 2017 - с 86-89
4. Сравнение Linux и Windows / Ravesli [Электронный ресурс] – URL: <https://ravesli.com/sravnenie-linux-i-windows-v-chem-raznitsa-i-chto-luchshe/>
5. Database Speed Comparision [Электронный ресурс] – URL: <https://www.sqlite.org/speed.html> (дата обращения 8.11.2021)
6. Справочник по HTML [Электронный ресурс]. – URL: http://htmlbook.ru/html (дата обращения: 10.11.2021)
7. 10 причин, почему мы перешли на PyCharm / Хабр [Электронный ресурс] – URL: https://habr.com/ru/post/122018/ (дата обращения: 1. 11.2021)
8. Г. С. Иванова, Т. Н. Ничушкина, Е.К. Пугачев, Р.С. Самарев. Методические указания по выполнению курсовой работы по дисциплине «Технология разработки программных систем». МГТУ имени Н.Э. Баумана. 2014. 38 c.
9. ГОСТ 7.32-2017. Межгосударственный стандарт. Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления. – [введен в действие Приказом Росстандарта от 24.10.2017 N 1494-ст]