МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К. И. Сатпаева

Институт автоматики и информационных технологий

Кафедра "Программная инженерия"

ДОПУЩЕН К ЗАЩИТЕ
Заведующая кафедрой ПИ
канд. физ-мат. наук, профессор
_____ А.Н. Молдагулова
«20» 05 2022 г.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к дипломному проекту

На тему: “Разработка веб-приложения, для написания музыки”

по специальности 5В070400 – Вычислительная техника и программное обеспечение

Выполнил Мамедов А. М

Рецензент
Доктор PhD, доцент
_____ А. У. Утегенова
«20» 05 2022 г.

Научный руководитель
Доктор PhD
Ассоциированный профессор
_____ М. Жекамбаева
«20» 05 2022 г

Алматы 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К. И. Сатпаева

Институт автоматики и информационных технологий

Кафедра "Программная инженерия"

5В070400 – Вычислительная техника и программное обеспечение

УТВЕРЖДАЮ
Заведующая кафедрой ПИ
канд. физ-мат. наук, профессор
_____ А.Н. Молдагулова
«20» 05 2022 г.

ЗАДАНИЕ
на выполнение дипломного проекта

Обучающемуся *Мамедову Асдану Мирза Оглы*
Тема: *Разработка веб-приложения, для написания музыки*
Утверждена *приказом проректора по академической работе* №489-1710 от «24» 12 2021 г.
Срок сдачи законченного проекта «26» 05 2022 г
Исходные данные к дипломному проекту: *техническое задание, описание БД в виде ER-диаграммы, описание нужных функций проекта.*
Перечень подлежащих разработке в дипломном проекте вопросов:
а) разработка дизайна к веб-приложению;
б) создание реляционные модели базы данных;
в)разработка бэк-энд части на фрейворке языка программирования Python «Django»
г) тестирование, отладка, добавление на хостинг
Перечень графического материала (с точным указанием обязательных чертежей): *представлены 22 слайда презентации.*
Рекомендуемая основная литература: *из 20 наименований.*

ГРАФИК

подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю и консультантам	Примечание
1. Анализ предметной области, разработка технического задания	17.01.2022	Выполнено
2. Выбор технологий для разработки	22.01.2022	Выполнено
3. Разработка базы данных	27.01.2022	Выполнено
4. Разработки логики взаимодействия	01.02.2022	Выполнено
5. Разработка функционала системы	07.02.2022	Выполнено
6. Тестирование и оптимизация	14.02.2022	Выполнено
7. Написание пояснительной записки к дипломному проекту	29.04.2022	Выполнено

Подписи

консультантов и нормоконтролера на законченный дипломный проект с указанием относящихся к ним разделов проекта

Наименования разделов	Консультанты, И.О.Ф. (уч. степень, звание)	Дата подписания	Подпись
Нормоконтролер	Жекамбаева М.Н. Доктор Ph.D., ассоциированный профессор	20.05.22	
Программное обеспечение	Марғұлан Қ. Магистр техн.наук, лектор	18.05.22	

Научный руководитель _____ Жекамбаева М.

Задание принял к исполнению обучающийся _____ Мамедов А. М.

Дата «17» 11 2021г.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет
имени К. И. Сатпаева

Институт автоматики и информационных технологий

Кафедра "Программная инженерия"

Мамедов Асдан Мирзаоглы

Разработка веб-приложения, для написания музыки

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к дипломному проекту

5В070400 – Вычислительная техника и программное обеспечение

Алматы 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет
имени К. И. Сатпаева

Институт автоматики и информационных технологий

Кафедра "Программная инженерия"

ДОПУЩЕН К ЗАЩИТЕ
Заведующая кафедрой ПИ
канд. физ-мат. наук, профессор
_____ А.Н. Молдагулова
«_____» _____ 2022 г.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к дипломному проекту

На тему: “Разработка веб-приложения, для написания музыки”

по специальности 5В070400 – Вычислительная техника и программное обеспечение

Выполнил Мамедов А. М

Рецензент
Доктор PhD, доцент

_____ А. У. Утегенова
«____»_____ 2022 г.

Научный руководитель
Доктор PhD
Ассоциированный профессор
_____М. Жекамбаева
«___» _____ _____ 2022 г

Алматы 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К. И. Сатпаева

Институт автоматики и информационных технологий

Кафедра "Программная инженерия"

5В070400 – Вычислительная техника и программное обеспечение

УТВЕРЖДАЮ
Заведующая кафедрой ПИ
канд. физ-мат. наук, профессор
_____ А.Н. Молдагулова
«_____» _____ 2022 г.

ЗАДАНИЕ
на выполнение дипломного проекта

Обучающемуся *Мамедову Асдану Мирза Оглы*
Тема: *Разработка веб-приложения, для написания музыки*
Утверждена *приказом проректора по академической работе №_____ от «___» _____ 20___ г.*
Срок сдачи законченного проекта *«___» _____ 20___ г*
Исходные данные к дипломному проекту: *техническое задание, описание БД в виде ER-диаграммы, описание нужных функций проекта.*
Перечень подлежащих разработке в дипломном проекте вопросов:
а) разработка дизайна к веб-приложению;
б) создание реляционные модели базы данных;
в)разработка бэк-энд части на фрейворке языка программирования Python «Django»
г) тестирование, отладка, добавление на хостинг
Перечень графического материала (с точным указанием обязательных чертежей): *представлены 22 слайда презентации.*
Рекомендуемая основная литература: *из 20 наименований.*

ГРАФИК

подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	**Сроки представления научному руководителю и консультантам**	**Примечание**
1. Анализ предметной области, разработка технического задания	17.01.2022	Выполнено
2. Выбор технологий для разработки	22.01.2022	Выполнено
3. Разработка базы данных	27.01.2022	Выполнено
4. Разработки логики взаимодействия	01.02.2022	Выполнено
5. Разработка функционала системы	07.02.2022	Выполнено
6. Тестирование и оптимизация	14.02.2022	Выполнено
7. Написание пояснительной записки к дипломному проекту	29.04.2022	Выполнено

Подписи

консультантов и нормоконтролера на законченный дипломный проект с указанием относящихся к ним разделов проекта

Наименования разделов	**Консультанты, И.О.Ф. (уч. степень, звание)**	**Дата подписания**	**Подпись**
Нормоконтролер	Жекамбаева М.Н. Доктор Ph.D., ассоциированный профессор		
Программное обеспечение	Марғұлан Қ. Магистр техн.наук, лектор		

Научный руководитель _____ Жекамбаева М. Н.

Задание принял к исполнению обучающийся _____ Мамедов А. М.

Дата «___» _____ 2021г.

АННОТАЦИЯ

Данный дипломный проект посвящен разработке веб-приложения, для написания музыки, используя способности языков программирования и библиотек.

За всю цивилизованную историю, музыка является неотъемлемой частью нашей жизни. В современном мире с появлением технологии, каждый имеет доступ к музыке, а также и к написанию его. С появлением интернета человек имеющий доступ к компьютеру может написать свою музыку. В Казахстане, с каждым днем все больше и больше людей, хотят научиться создавать свои песни. Веб-приложение будет содержать курс, который поможет написать свое первое произведение.

Для верстки дизайна используется Bootstrap — это самый популярный CSS Фреймворк для разработки отзывчивых и мобильных веб-приложений. Для хранения и обработки информации используются способности реляционной база данных SQLite, которая встроена в Python и язык запросов SQL для взаимодействия с базой данных. Для разработки back-end используется Фреймворк языка программирования Python «Django», который отвечает за серверную часть и запросы.

АНДАТПА

Бұл дипломдық жоба бағдарламалау тілдері мен кітапханалардың қабілеттерін қолдана отырып, музыка жазуға арналған веб-қосымшаны жасауға арналған.

Бүкіл өркениетті тарихта музыка біздің өміріміздің ажырамас бөлігі болып табылады. Қазіргі әлемде технологияның пайда болуымен әркімнің музыкаға, сонымен қатар оны жазуға мүмкіндігі бар. Интернеттің пайда болуымен компьютерге қол жеткізе алатын адам өз музыкасын жаза алады. Қазақстанда күн сайын көбірек адамдар өз әндерін жасауды үйренгісі келеді. Веб-қосымшада сіздің алғашқы жұмысыңызды жазуға көмектесетін курс болады.

Дизайн макеті үшін Bootstrap қолданылады-бұл жауап беретін және мобильді веб-қосымшаларды жасауға арналған ең танымал CSS негізі. Ақпаратты сақтау және өңдеу үшін SQLite реляциялық дерекқорының қабілеттері қолданылады, ол Python-ға және SQL сұрау тіліне мәліметтер базасымен өзара әрекеттесу үшін енгізілген. Артқы жағын жасау үшін Python бағдарламалау тілінің "Django" жақтауы қолданылады, ол сервер мен сұрауларға жауап береді.

ANNOTATION

This diploma project is dedicated to the development of a web application for writing music using the abilities of programming languages and libraries.

For the entire civilized history, music has been an integral part of our lives. In the modern world with the advent of technology, everyone has access to music, as well as to writing it. With the advent of the Internet, a person with access to a computer can write their own music. In Kazakhstan, every day more and more people want to learn how to create their own songs. The web application will contain a course that will help you write your first work.

Bootstrap is used for the layout of the design — it is the most popular CSS Framework for the development of responsive and mobile web applications. To store and process information, the capabilities of the SQLite relational database are used, which is built into Python and the SQL query language for interacting with the database. For back-end development, the Python programming language framework "Django" is used, which is responsible for the server side and queries.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Мир технологий развивается с быстрой скоростью. В прошлом веке чтобы найти и изучить информацию, нужно было сильно постараться. Ходить по библиотекам, записываться на специальные курсы, которые находятся в разных городах и странах.

Порог вхождение в мир музыки был очень велик, обычный житель не мог себе позволить сочинить музыку у себя дома и выпустить ее. Нужно не малое количество денег вложить в обучения и оборудования чтобы стать успешным в своем русле.

С появлением интернета информация стала доступна почти в каждом доме. Любой житель может теперь найти книгу, информацию, которая поможет ему познать определенное русло, мир музыки не исключение. Так же теперь можно писать музыку просто имея обычный ноутбук.

В данную ситуацию в интернете много информации, достоверной и неправильной. Не всегда пользователь может найти знания, который даст толчок в развитии написании музыки. Веб-приложение создана, чтоб собрать всю правильную информацию в один курс и сделать обучение с большой отдачей.

Хотелось бы протестировать возможно ли учение с помощью веб приложения и видеоуроков или нет. Так же приложение имеет удобный интерфейс как для ученика, так и для учителя.

1 Исследовательский раздел

1.1 Цель разработки

Основной целью этого проекта, разработать веб-приложение, для написания музыки, который поможет любителям музыки, научиться писать песни.

Веб-приложение представляет собой курс, где есть определенные уроки, а также домашние задания. Преимущества такого курса заключается в структурированном обучением, а также удобством интерфейса. Клиентская часть создана в минималистичном стиле, которая позволяет музыкантам быстро освоиться в приложении.

1.2 Определения, термины и сокращения

В таблице 1.1 сформулированы все термины и сокращения, которые используются в предметной области разрабатываемого проекта, а также специфические термины, связанные с программной реализацией проекта и используемыми технологиями при разработке

Таблица-1.1 – Сокращения, термины и их определения

Сокращение или термин	Определение
СУБД	Система управления базами данных
БД	База Данных
HTML	Hypertext Markup Language. Язык гипертекстовой разметки.
CSS	Cascading Style Sheets
JS	JavaScript – Язык программирования
Фреймворк	платформа для создания сайтов и приложений, которая облегчает разработку благодаря большому количеству реализованных функций
Visual Studio Code	Редактор кода
UML	Unified Modeling Language - Унифицированный язык моделирования
ER	Entity Relationship – модель данных
SQL	Structured Query Language – язык программирования для управления данными в реляционной базе данных

Продолжение таблицы-1.1 - Сокращения, термины и их определения

API	Application Programming Interface
CRUD	Create, Read, Update, Delete operations
MVT	Model View Template
MVC	Model View Controller
DRY	Don’t Repeat Yourself
URL	Universal Resource Locator
GET	Hypertext Transfer Protocol Method
POST	Hypertext Transfer Protocol Method
ORM	Object Relational Mapping

1.3 Актуальность проблемы на момент написания дипломный работы

Проблема написании музыки на данный момент, заключается в том, что нет в интернете определенной структуры по обучению. У новичка есть шанс наткнуться на неправильный по содержанию и по его уровню урок, который понизит интерес и оттолкнет от изучения. Отсутствие домашних заданий, замедляет обучение, и не каждый ученик будет оттачивать навыки, а переходить на следующий урок.

Веб-приложение прежде всего, имеет пошаговое обучение как показано на рисунке 1.1, который позволит ученикам не запутаться в уроках, и не оттолкнет от изучения материала.

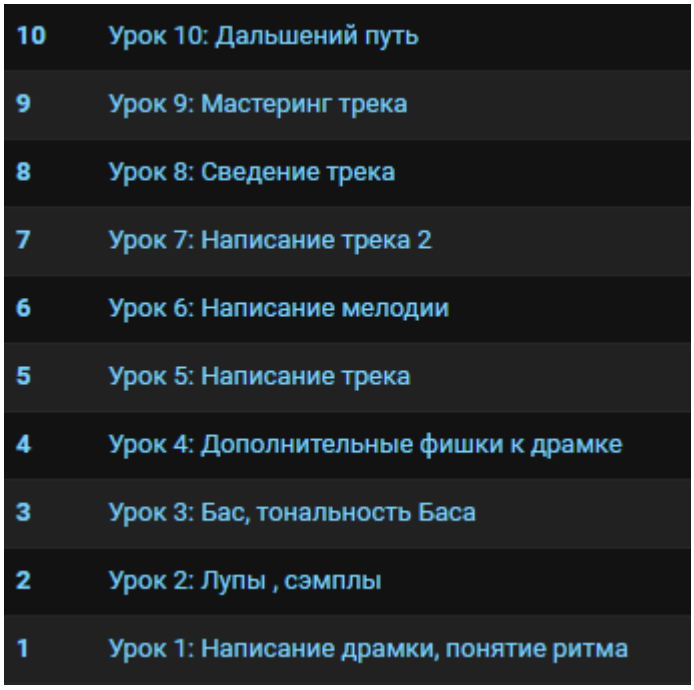

Рисунок-1.1 – Список уроков

Также после каждого урока будет задана домашняя работа, для того, чтоб, ученик смог применить, то, что будет изучать на уроках и оттачивать навыки.

2 Технологический раздел

2.1 Bootstrap

Bootstrap[1] - это бесплатный CSS-фреймворк с открытым исходным кодом, предназначенный для адаптивной, ориентированной на мобильные устройства интерфейсной веб-разработки. Он содержит HTML[2], CSS[3] как показано на рисунке 2.1 и шаблоны дизайна на основе JavaScript[4] для типографики, форм, кнопок, навигации и других компонентов интерфейса.

Благодаря Bootstrap можно быстро сверстать минималистичный дизайн для клиентской части веб-приложения.

```
<div class="card asdanmusic" style="max-width: 100%">
  <div class="row">
    <div class="col col-md-4 asdanmusic-img">
      <img src="{% static 'img/asdanmusic.png' %}"  class="asdanmusic-img__photo">
    </div>
    <div class="col-md-8">
      <div class="card-body">
        <h5 class="card-title">Кто я ?</h5>
        <p class="card-text asdanmusic__p">Меня зовут Асдан. Я музыкант, битмейкер
      </div>
    </div>
  </div>
</div>
```

Рисунок-2.1 – классы Bootstrap в верстке

2.1.1 Среда разработки

Среда разработки программного обеспечения и веб-разработки — это рабочее пространство, в котором разработчики могут вносить изменения, ничего не нарушая в реальной среде.

Основная цель использования - писать код.

Visual Studio Code[5] - это бесплатный редактор, который помогает программисту писать код, помогает в отладке и исправляет код. В обычных условиях это облегчает пользователям написание кода простым способом.

В данной работе использую эту среду разработки, главными преимуществами являются, помощники кода с помощью аддонов и скорость самого приложения.

2.2 Python Django

Django[6] - это высокоуровневый веб-фреймворк на Python, который поощряет быструю разработку и чистый, прагматичный дизайн. Созданный опытными разработчиками, он берет на себя большую часть хлопот, связанных с веб-разработкой, так что можно сосредоточиться на написании своего приложения без необходимости изобретать велосипед. Это бесплатно и с открытым исходным кодом.

2.2.1 Преимущества Python Django

Один из главных преимуществ веб-фреймворка заключается в том, что код не повторяется, все из-за принципа DRY[7], это означает написанная концепция должна находиться на одном месте. Также оптимизированы SQL запросы.

Работает по архитектуре MVT как показано на рисунке 2.2, пользователь с помощью URL - адреса отправляет запрос на сервер, происходит обработка, подбирается модель данных и клиентский шаблон, в котором будет показан результат[8].

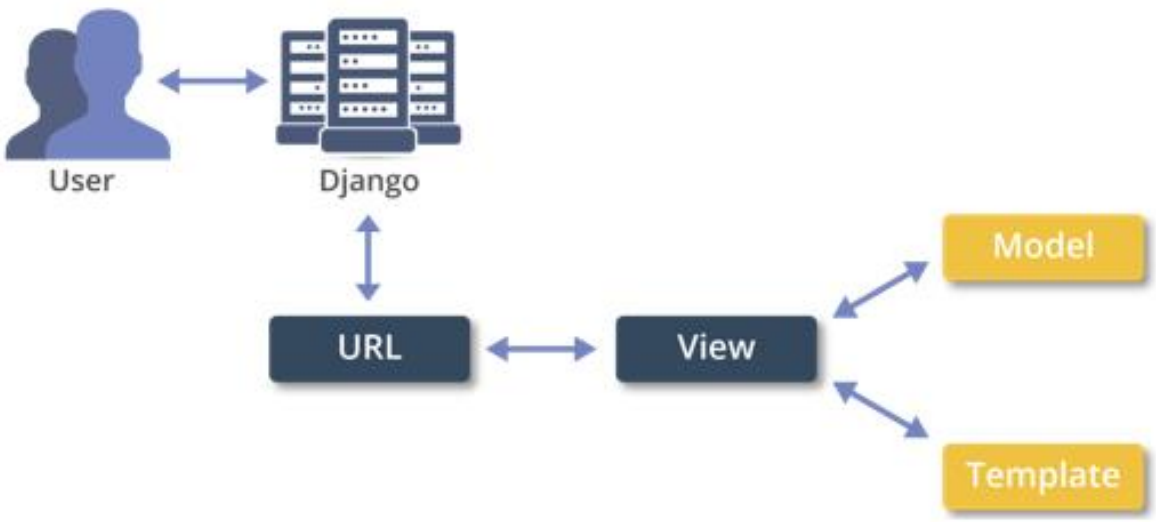

Рисунок-2.2 – Model View Template

Django в плане безопасности занимает высокое место в категории Back-End частей. Помогает избежать часто распространенные ошибки безопасности, включая, межсайтовые подделки запросов(csrf), скриптов и т. д.

Фреймворк поддерживает такие базы данных, как PostgreSQL, MySQL, SQLite[9] и т.д.

2.2.2 Функции Python Django

В Django представления(views) выступают в качестве связующего звена между данными модели и шаблонами. Так же, как и контроллер в MVC[10], представления в Django MVT отвечают за обработку всей бизнес-логики, лежащей в основе веб-приложения. Он действует как мост между моделями и шаблонами. Он видит запрос пользователя, извлекает соответствующие данные из базы данных, затем возвращает шаблон вместе с полученными данными.

В данном проекте использовалось два типа представления это Class-based-views[11] и function views. В первом случае дается класс которые имеет заготовленные методы и свойства под запросы GET и POST[12]. Остается вписать правильно данные. Во втором случаем все прописываем сами. Пример из проекта как показано на рисунке 2.3.

```
class LessonView(DetailView): ## класс подкласс detailview
    #чтоб рассмотреть элемент из сэта, туда пишем модель и с помощью встроенного аргумента pk
    ## можем рассмотреть отдельный элемент
    model = Lessons
    template_name = 'course/lesson.html'
    context_object_name = 'lesson'
```

Рисунок-2.3 – Class-Based-View

3 Проектная часть

3.1 Архитектура проекта

Веб-приложение разработано на основе стека Django, сюда входит HTML, CSS, JavaScript, Bootstrap, Python Django, SQLite как показано на рисунке 3.1.

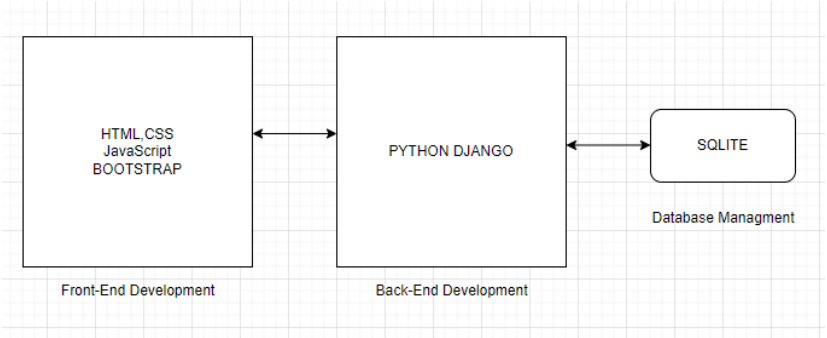

Рисунок-3.1 – Стек для разработки веб-приложения

Все записанные данные должны храниться в реляционной базе данных SQLite. С помощью веб-фреймворка Python Django, получаем возможность добавлять, удалять, считывать и редактировать данные из базы данных. К модели данных подключаем нашу базу данных. Происходит передача из модели в представления, и все это упаковывается в шаблоне, который будет показан клиенту. Чтобы получить из модели данных данные прописываем в представлениях ORM[13] запросы, которые пишутся на языке Python, а не SQL, что упрощает нам работу с данными. Шаблоны написаны на принципе DRY, есть один родительский шаблон, и в новых файлов мы дополняем код с помощью Django Template Tags[14] как показано на рисунке 3.2.

```
{% extends "base/base.html" %}
{% load static %}

{% block title %}
{{title}} :: {{block.super}}
{% endblock title %}
{% block content %}
<!-- проверка логина , учитель,ученик или не зарегистрировавшийся пользователь -->
{% if request.user.is_authenticated %}


        {% if user.is_superuser %}
            {% include "login_content_super_user/_login_content.html" %}
        {% else %}
            {% include "login_content_user/_login_content.html" %}
        {% endif %}

{% else %}


    {% include "unlogin_content/_unlogin_content.html" %}

{% endif %}

{% endblock content %}
```

Рисунок-3.2 – Template Tags Python Django

3.2 Сценарий Использования

Это письменное описание того, как пользователи будут выполнять задачи на веб-приложении. Описывает, с точки зрения пользователя, поведение системы при ответе на запрос. Каждый вариант использования представлен в виде последовательности простых шагов, начинающихся с цели пользователя и заканчивающихся, когда эта цель достигнута[15]. На рисунке 3.3 представлен сценарий использования.

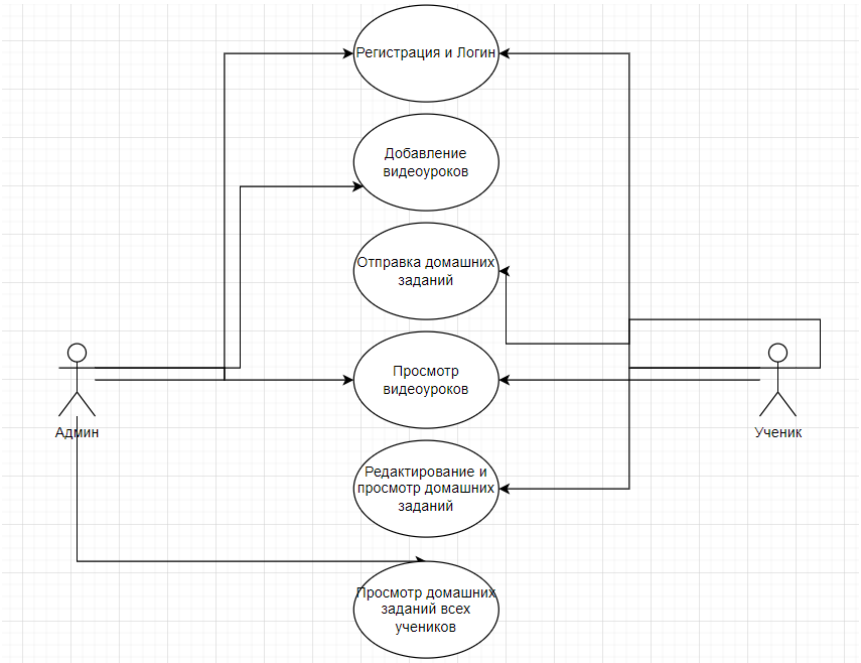

Рисунок-3.3 – Сценарий использования веб-приложения

Два главных персонажа — это Админ и Ученик. У каждого свои полномочии.

Регистрация и логин – каждый пользователь должен зарегистрироваться, чтобы смотреть уроки.

Добавление видеоуроков – Админ является учителем, и должен добавлять видеоуроки через удобный с помощью Python Django личный кабинет, который имею простой интерфейс как показано на рисунке 3.4.

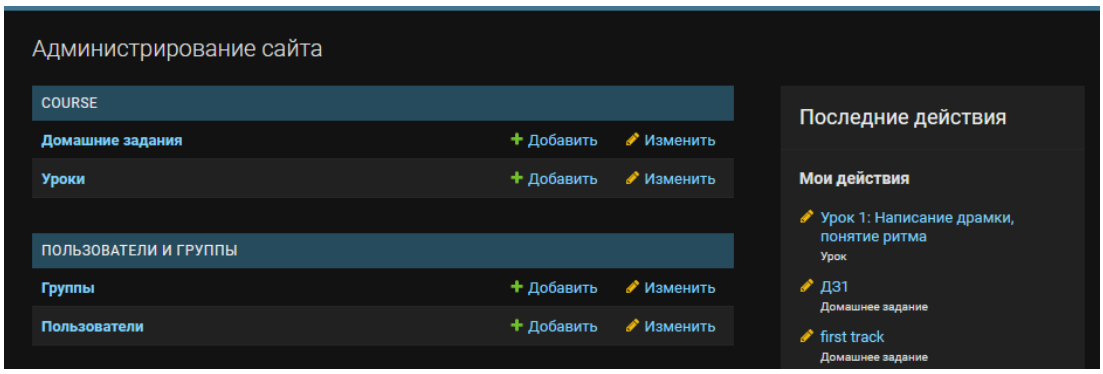

Рисунок-3.4 – Интерфейс личного кабинета админа

Отправка домашних заданий – Ученик отправляет после каждого урока домашнее задание, в описании урока есть кнопка, которая перекидывает на форму отправки задания, как показано на рисунке 3.5

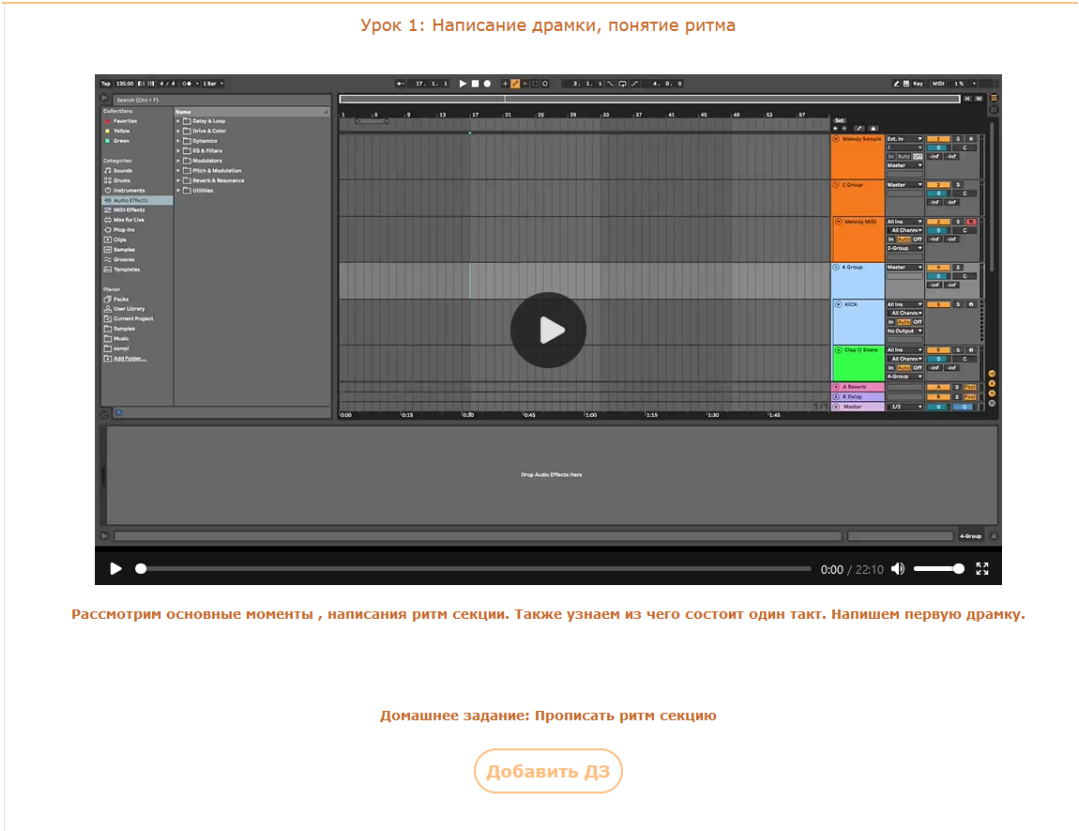

Рисунок-3.5 – Отправка задания

Просмотр видеоуроков – после добавление уроков, в кабинете у ученика, пополняется список уроков, с каждой неделей добавляется один урок. Показывается описание, сам урок и домашнее задание.

Редактирование и просмотр домашних заданий – Ученик может редактировать и смотреть домашние задания только свои как показано на рисунке 3.6.

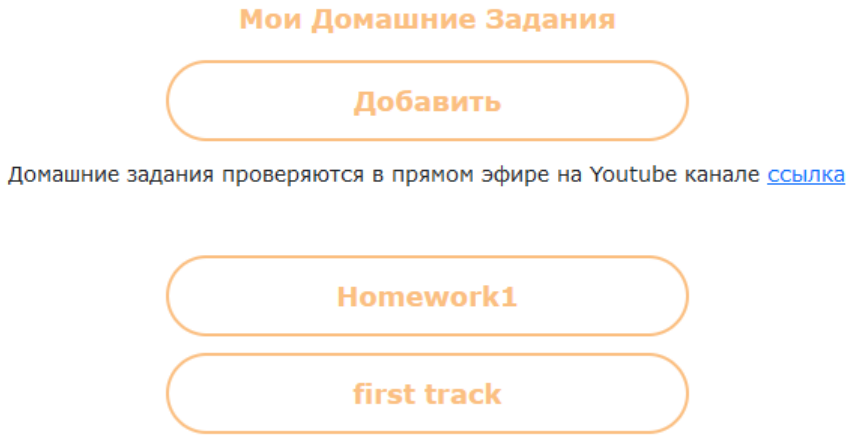

Рисунок-3.6 – Список домашних заданий

Просмотр домашних заданий всех учеников – В кабинете учителя все задания учеников сортированы по определенному уроку, это оптимизирует проверку и облегчит работу как показано на рисунке 3.7.

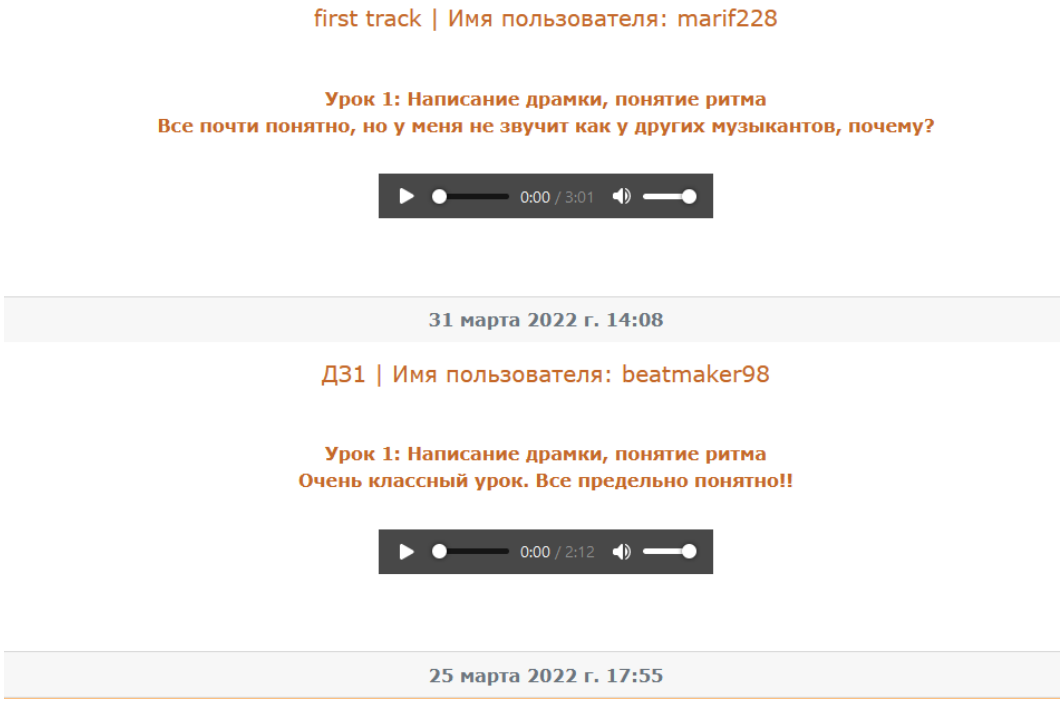

Рисунок-3.7 – Список заданий учеников

3.3 Разработка серверной части

Все работа с данными, логика веб-приложения происходят на серверной части.

Работа будет вестись с помощью веб-фреймворка Python Django, используя концепцию MVT[16]. Модель данных разрабатывается на основе базы данных. Сам Django является контроллером, который получает URL-запрос[17] и отправляет его в представления. Модель данных поступает в представление, который упакует данные в определенный шаблон. Используются CRUD операции[18].

3.3.1 Регистрация и авторизация

Один из плюсов Django, поставляется с системой аутентификации пользователя. Он обрабатывает учетные записи пользователей, группы, разрешения и пользовательские сеансы. Так же имеет встроенную базу данных, которая упрощает работу и не нужно создавать всю логику регистрации и авторизации. Все что надо, это импортировать в проект и подключить через настройки как показано на рисунке 3.8.

```
# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'ckeditor',
    'debug_toolbar',
    'course.apps.CourseConfig',

]
```

Рисунок-3.8 – Установка авторизации

Для создания формы регистрации импортируем модель данных «User», после этого настраиваем формочку как показано на рисунке 3.9.

```python
class UserRegisterForm(UserCreationForm):
    username = forms.CharField(label='Имя пользователя',
    widget= forms.TextInput(attrs={'class':'form-control',}))
    password1 = forms.CharField(label='Пароль',
    widget= forms.PasswordInput(attrs={'class':'form-control'}))
    password2 = forms.CharField(label='Подтверждение пароля',
     widget= forms.PasswordInput(attrs={'class':'form-control'}))
    email = forms.EmailField(label='Электронный адрес',
    widget= forms.EmailInput(attrs={'class':'form-control'}))

    class Meta:
        model = User
        fields = ('username', 'email' , 'password1', 'password2')
```

Рисунок-3.9 – Форма регистрации

Таким способом настраиваем форму для авторизации, хватит всего два поля это логин и пароль как показано на рисунке 3.10.

```python
class UserLoginForm(AuthenticationForm):
    username = forms.CharField(label='Имя пользователя',
     widget= forms.TextInput(attrs={'class':'form-control'}))
    password = forms.CharField(label='Пароль', widget=
    forms.PasswordInput(attrs={'class':'form-control'}))
```

Рисунок-3.10 – Форма авторизации

Чтобы формы заработали в главном контроллере прописываем логику, которая будет передаваться в шаблон с помощью «HTTP» методов, будет определяться отправка или показ формы как показано на рисунке 3.11.

```python
def Home(request):
    if request.method == 'POST':
        form = UserRegisterForm(request.POST)
        if form.is_valid():
            user = form.save()
            login(request,user)
            messages.success(request,'Успешная регистрация')
            redirect('Home')
        else:
            messages.error(request, 'Ошибка')
    else:
        form = UserRegisterForm()
    return render(request,'course/index.html',
    {"form" : form , "title" : "Главная Страница"})
```

Рисунок-3.11 – Регистрация «View»

Если данные проходят по всем правилам валидатора, то пользователь зачисляется в базу данных. Для авторизации используется аналогичный принцип, только без сохранения новых данных.

```
def user_login(request):
    if request.method == 'POST':
        form = UserLoginForm(data = request.POST)
        if form.is_valid():
            user = form.get_user()
            login(request, user)
            return redirect('Home')
    else:
            form = UserLoginForm()
    return render(request, 'course/login.html',
    {"form" : form, "title" : "Вход"})
```

Рисунок-3.12 – Логин «view»

Ответы передаются в шаблон и создается верстка формы как показано на рисунке 3.13.

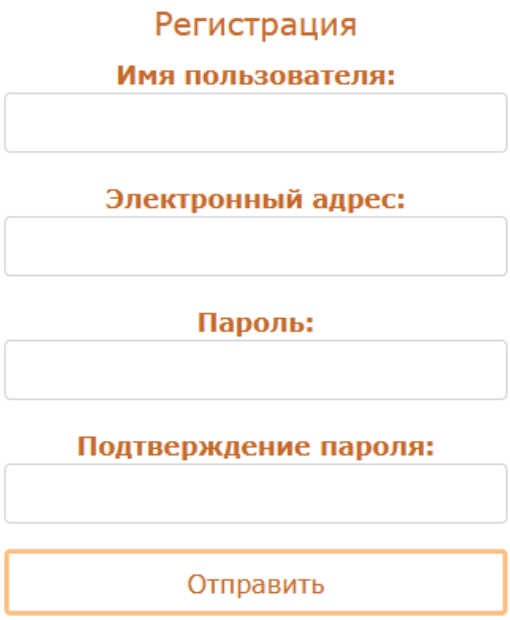

Рисунок-3.13 – Форма регистрации в шаблоне

Также веб-фреймворк имеет csrf_token, которые предоставляют защиту против Межсайтовой подделки запроса. Этот тип атак случается, когда злонамеренный веб-сайт содержит ссылку, кнопку формы или некоторый javaScript, который предназначен для выполнения некоторых действий на вашем Web сайте, используя учетные данные авторизованного пользователя, который посещал злонамеренный сайт в своем браузере[19].

3.3.2 База данных SQLite

SQLite — это библиотека на языке C, которая реализует небольшой, быстрый, автономный, высоконадежный, полнофункциональный движок базы данных SQL. Наиболее часто используемый движок баз данных в мире. Встроен во все мобильные телефоны и большинство компьютеров и поставляется в комплекте с бесчисленным множеством других приложений, которые люди используют каждый день.

В этом проекте как СУБД, выбран SQLite. С помощью его просто разработать реляционные базы данных, не нужно настраивать. Так же SQLite по умолчанию стоит в серверной части веб-приложения. На рисунке 3.14 показана ER-диаграмма[20].

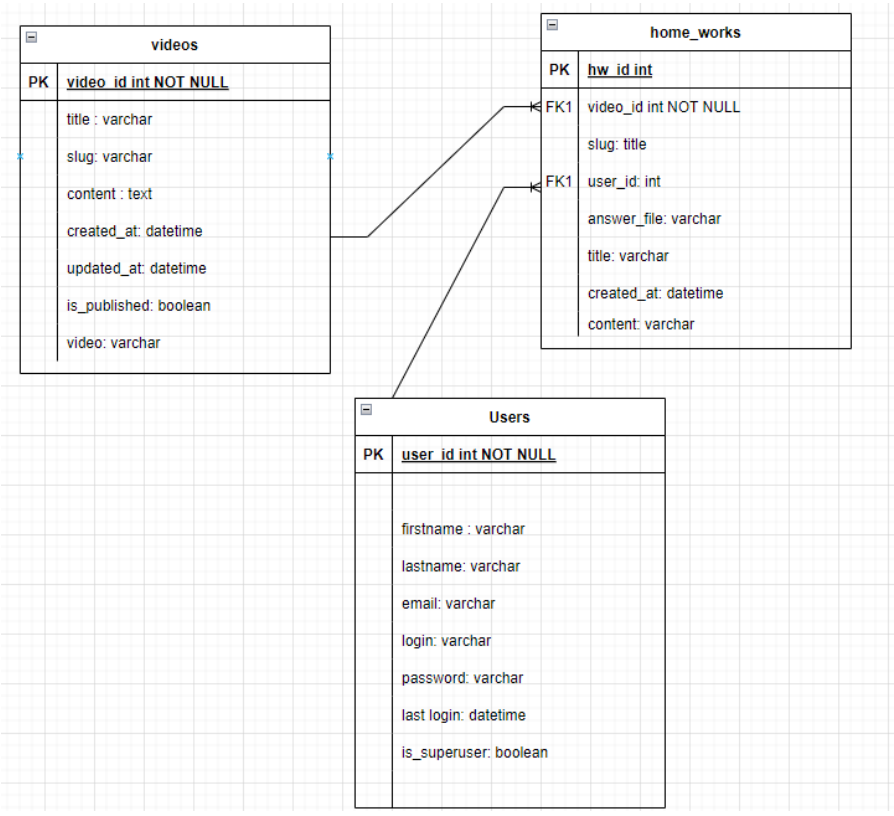

Рисунок-3.14 – ER – диаграмма

Чтобы связать базу данных с серверной частью, нужно прописать модель данных и провести миграцию, таблицы автоматически появятся в базе данных.

Таблица «Home Works» - главная таблица на этом курсе, так как содержит домашние задания учеников. Главными полями считается «video_id» и «user_id», с помощью них можно узнать, кто отправил задание и на какое видео ссылается. В поле «answer_file» записано адрес местонахождение задания, при создании базы данных, прописывается путь. Остальные поля вспомогательные. Первичным ключом является поле «home_work_id»

Таблица «Videos» - таблица для добавления видеоуроков. Так же при разработке указывается путь, где будут храниться файлы. Первичным ключом является поле «video_id».

Таблица «Users» - предназначена для регистрации пользователей. В себе хранит главные поля для правильной работы системы. Первичным ключом является «user_id», также является вторичным ключом для таблицы «Home Works». Самыми главными полями являются «email», «login», «password». Также не менее важным является «last_login», служит для развития функционала и удобств веб-приложения и «is_superuser», который нужен чтоб различать ученика с учителем. Так как у них полномочии разные. Учитель может просматривать все отправленные задания учеников, а студент всего лишь свои файлы. Так же благодаря правильной выстроенной таблицы, домашние задания сортируются по урокам. Это облегчает работу и повышает производительность.

Желательно сначала представить или нарисовать диаграмму, как будет выглядеть база данных до внедрения в проект. Так как легче будет начать сначала, чем исправлять ошибки

ЗАКЛЮЧЕНИЕ

Результатом данной дипломной работы является веб-приложение по созданию музыки «ASDANMUSIC», которая содержит курс по музыке. Веб-приложение ориентированно на людей, которые хотят начать путь в музыкальной карьере. Основными требованиями были функциональные возможности:

- регистрация и авторизация;
- реализация CRUD операций, таких как:
    - добавление уроков;
    - редактирование уроков;
    - удаление уроков;
    - просмотр уроков;
    - добавление домашнего задания;
    - просмотр домашнего задания;
    - редактирование домашнего задания;
    - удаление домашнего задания;

Все поставленные задачи удалось реализовать. Так как реализовано на python Django, функционал веб-приложения с легкостью масштабировать. Проект был размещен на хостинг. В процессе разработки были улучшены навыки работы в Back-End и управлением реляционный базы данных. Проект был правильно разделен на бизнес логику и логику отображения.

С помощью данного курса будет начальное представление, которая поможет изучающему, выбрать свое направление в музыке. Так же видеоуроки всегда останутся с ними, и студент может всегда вернуться и пересмотреть материал.

Все аккаунты данного веб-приложения защищены с помощью «csrf_token», который шифрует отправленное через форму, и посторонние мошеннические сайты не могут использовать данную форму для привлечение других клиентов к обману.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1 Документация Bootstrap // Электронная версия на сайте https://getbootstrap.com/docs/5.1/getting-started/introduction/

2 Документация HTML // Электронная версия на сайте https://developer.mozilla.org/ru/docs/Web/HTML

3 Документация CSS // Электронная версия на сайте https://developer.mozilla.org/ru/docs/Web/CSS

4 Документация JavaScript // Электронная версия на сайте https://developer.mozilla.org/ru/docs/Web/JavaScript

5 Документация Visual studio code // Электронная версия на сайте https://code.visualstudio.com/docs

6 Документация Python Django // Электронная версия на сайте https://docs.djangoproject.com/en/4.0/

7 Django: DRY и названия полей моделей в шаблонах // Игорь Стариков // Статья https://pythonz.net/articles/16

8 Принципы Python Django // Электронная версия на сайте // Cordenne Brewster https://trio.dev/blog/what-is-django

9 Документация SQLite // Электронная версия на сайте https://www.sqlite.org/docs.html

10 MVC Документация // Электронная версия на сайте https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm

11 Class-Based-Views // Электронная версия на сайте https://docs.djangoproject.com/en/4.0/topics/class-based-views/

12 HTTP методы документация // Электронная версия на сайте https://developer.mozilla.org/ru/docs/Web/HTTP/Methods/POST

13 ORM запросы документация // Электронная версия на сайте https://docs.djangoproject.com/en/4.0/topics/db/queries/

14 Django Template Tags // Электронная версия на сайте https://docs.djangoproject.com/en/4.0/ref/templates/builtins/

15 What is Use Case Diagram? // Электронная версия на сайте https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/

16 MVT концепция // Электронная версия на сайте https://www.geeksforgeeks.org/django-project-mvt-structure/

17 URL документация // Электронная версия на сайте https://developer.mozilla.org/ru/docs/Web/API/URL

18 What is CRUD // Электронная версия на сайте https://www.codecademy.com/article/what-is-crud

19 Подделка межсайтового запроса // Электронная версия на сайте https://docs.djangoproject.com/en/4.0/ref/csrf/

20 What is Entity Relationship Diagram? // Электронная версия на сайте https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/

Приложение А
(обязательное)

Техническое задание

А.1.1 Техническое задание на разработку веб-приложения по написанию музыки

Настоящее техническое задание распространяется на разработку веб-приложение по написанию музыки, предназначенной для обучения любителей музыки и творчества. Предполагаются, что в данной системе будут учителя и ученики. Автоматизация обучения позволит улучшить качество и производительность уроков и получения навыков для студентов.

А.1.1.1 Основание для разработки

Веб-приложение разрабатывается на основе устного согласия научного руководителя выбранной дипломником темой.

А.1.1.2 Назначение

Веб-приложение назначена для обучения по написанию по музыке, также для хранения, сбора и обработки информации.

А.1.1.3 Требования к функциональным характеристикам

С помощью языка Python и веб-фреймворка Django нужно создать веб-приложение по написанию музыки.

У пользователей данного веб-приложения есть роли: учитель (администратор) и студент (обычный пользователь). Учитель имеет доступ к административному сайту, к базе данных, может добавлять, изменять и удалять уроки. Также проверять все домашние задания учеников. Студент может смотреть уроки и добавлять, изменять и удалять свои домашние задания.

Должна использоваться реляционная база данных для хранения информацию об уроках, домашних заданий, и пользователей.

Должна быть размещена на хостинге, который обеспечит доступ в любое время.

Система должна обеспечить выполнение данных функций

- регистрация и логин пользователей;
- добавления, просмотр, изменение и удаление уроков;
- добавления, просмотр, изменение и удаление домашних заданий;
- просмотр всех домашних заданий для учителя;

А.1.1.4 Требования к надежности

Обеспечить оптимизацию работы, постоянное тестирование на ошибки, применить «csrf_token», для защиты данных при заполнении форм.

Приложение Б
(обязательное)

Текст программы

Ниже приведен код основных, особенно важных функций.

1. Models.py – модель данных

```
from django.db import models
from django.contrib.auth.models import User
from django.shortcuts import redirect
from django.urls import reverse
from .utils import gen_slug


# Create your models here.

class Lessons(models.Model):
    title = models.CharField(max_length=150)
    slug = models.SlugField(max_length=255, verbose_name='Url', unique=
True)
    content = models.TextField(blank= True)
    created_at = models.DateField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now= True)
    video = models.FileField(upload_to='static/videos/%Y/')
    is_published = models.BooleanField(default=True)

    def __str__(self):
        return self.title

    class Meta:
        verbose_name = 'Урок'
        verbose_name_plural = 'Уроки'
        ordering = ['created_at']

    def get_absolute_url(self):
        return reverse("Lesson", kwargs={"slug": self.slug})
```

Продолжение приложения Б

```
class Home_Work(models.Model):
    title = models.CharField(max_length=150)
    slug = models.SlugField(max_length=255, verbose_name='Url', unique=
True, blank=True)
    content = models.TextField(blank= True)
    created_at =
models.DateTimeField(auto_now_add=True,verbose_name='Опубликовано')
    answer_file = models.FileField(upload_to='hworks/%Y/%m/%d/' , blank=
True)
    homeWork_check = models.BooleanField(default=False,
verbose_name='Проверка ДЗ')
    user = models.ForeignKey(User,on_delete=models.PROTECT,blank=True,
related_name='hworks')
    video = models.ForeignKey(Lessons,on_delete=models.PROTECT,
blank=True)

    def __str__(self):
        return self.title

    class Meta:
        verbose_name = 'Домашнее задание'
        verbose_name_plural = 'Домашние задания'
        ordering = ['-created_at']

    def get_absolute_url(self):
        return reverse("HomeWork", kwargs={"slug": self.slug })

    def save(self, *args,**kwargs):
        if not self.id:
            self.slug = gen_slug(self.title)
        return super().save(*args,**kwargs)
```

2. Admin.py – административный сайт для учителя

```
from django.contrib import admin
from django import forms
from ckeditor_uploader.widgets import CKEditorUploadingWidget

# Register your models here.

from .models import *
```

Продолжение приложения Б

```
class LessonsAdminForm(forms.ModelForm):
    content = forms.CharField(widget=CKEditorUploadingWidget())
    class Meta:
        model = Lessons
        fields = '__all__'

class LessonsAdmin(admin.ModelAdmin):
    prepopulated_fields = {"slug" : ("title",)}
    form = LessonsAdminForm

    save_on_top = True
    list_display = ('id','title', 'slug', 'video','created_at','updated_at', 'is_published',)
    list_display_links = ('id', 'title',)
    search_fields = ('title',)
    list_filter = ('is_published',)
    fields = ('title', 'slug', 'video','content','is_published',)


class Home_WorkAdmin(admin.ModelAdmin):
    prepopulated_fields = {"slug" : ("title",)}

    save_on_top = True
    list_display = ('id','title', 'slug', 'answer_file','created_at', 'user','video')
    list_display_links = ('id', 'title',)
    search_fields = ('title',)
    list_filter = ('video',)
    fields = ('title', 'slug','user','video', 'content','answer_file',)



admin.site.register(Lessons, LessonsAdmin)
admin.site.register(Home_Work, Home_WorkAdmin)
```

3. forms.py – логика форм

```
from django import forms

from django.contrib.auth.forms import UserCreationForm, AuthenticationForm
from django.contrib.auth.models import User
from django.core.exceptions import ValidationError
from .models import Home_Work
```

Продолжение приложения Б

```
class UserRegisterForm(UserCreationForm):
    username = forms.CharField(label='Имя пользователя',
    widget= forms.TextInput(attrs={'class':'form-control',}))
    password1 = forms.CharField(label='Пароль',
    widget= forms.PasswordInput(attrs={'class':'form-control'}))
    password2 = forms.CharField(label='Подтверждение пароля',
     widget= forms.PasswordInput(attrs={'class':'form-control'}))
    email = forms.EmailField(label='Электронный адрес',
    widget= forms.EmailInput(attrs={'class':'form-control'}))

    class Meta:
        model = User
        fields = ('username', 'email' , 'password1', 'password2')


class UserLoginForm(AuthenticationForm):
    username = forms.CharField(label='Имя пользователя',
     widget= forms.TextInput(attrs={'class':'form-control'}))
    password = forms.CharField(label='Пароль', widget=
    forms.PasswordInput(attrs={'class':'form-control'}))


class HomeWorkForm(forms.ModelForm):

    class Meta:
        model = Home_Work
        fields = ['title', 'content', 'video','answer_file']

        widgets = {
            'title': forms.TextInput(attrs={'class':'form-control'}),
            # 'slug': forms.TextInput(attrs={'class':'form-control'}),
            'content': forms.Textarea(attrs={'class' :'form-control'}),
            'video' : forms.Select(attrs={'class' : 'form-control'}),
            'answer_file' : forms.FileInput(attrs={'class' : 'form-control'}),

        }
    def clean_slug(self):
        new_slug = self.cleaned_data['slug'].lower()

        if new_slug == 'add':
            raise ValidationError('Назовите по другому')
        return new_slug
```

Продолжение приложения Б

4. urls.py – все url адреса

```
from unicodedata import name
from django import views
from django.urls import path, include

from .views import *

urlpatterns = [
    path('', Home,name = 'Home'),
    path('about/', About, name ='About'),
    path('lessons/', LessonsView.as_view(), name = 'Lessons'),
    path('lesson/<slug:slug>/', LessonView.as_view(), name = 'Lesson'),
    path('login/',user_login,name = 'Login'),
    path('logout/', user_logout,name = 'Logout'),
    path('homeworks/', HomeWorksView.as_view(), name = 'HomeWorks'),
    path('homework/add/', HomeWorkCreateView.as_view(), name
='HomeWork_add'),
    path('homework/<slug:slug>/', HomeWorkView.as_view(), name =
'HomeWork'),
    path('homework/<slug:slug>/update/', HomeWorkUpdateView.as_view(),
name = 'HomeWork_update'),
    path('homework/<slug:slug>/delete/', HomeWorkDeleteView.as_view(),
name = 'HomeWork_delete'),
    path('assignment/',AssignmentHome,name = 'AssignmentHome'),
    path('assignment/<int:video_id>/',
AssignmentVideo,name='AssignmentVideo'),
]
```

5. views.py – главная логика веб-приложения

```
from django.shortcuts import render, redirect
from django.http import HttpResponse
from django.views.generic import ListView,DetailView,CreateView,
UpdateView, DeleteView
from .models import *
from django.contrib.auth.models import User
from django.contrib import messages
from django.contrib.auth import authenticate, login, logout
from django.core.paginator import Paginator
from django.contrib.auth.models import User
from django.contrib.auth.mixins import LoginRequiredMixin
```

Продолжение приложения Б

```
from django.urls import reverse_lazy


from .forms import *

def Home(request):
    if request.method == 'POST':
        form = UserRegisterForm(request.POST)
        if form.is_valid():
            user = form.save()
            login(request,user)
            messages.success(request,'Успешная регистрация')
            redirect('Home')
        else:
            messages.error(request, 'Ошибка')
    else:
        form = UserRegisterForm()
    return render(request,'course/index.html',{"form" : form , "title" : "Главная Страница"})


def About(request):
    return render(request,'course/about.html')


class LessonsView(ListView):
    model = Lessons
    template_name = 'course/lessons.html'
    context_object_name = 'lessons'

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        context['title'] = 'Уроки'
        return context




    def get_queryset(self): ##метод получение queryset
        return Lessons.objects.filter(is_published = True) ##фильтрование
```

Продолжение приложения Б

```
class LessonView(DetailView): ## класс подкласс detailview
        #чтоб рассмотреть элемент из сэта, туда пишем модель и с помощью встроенного аргумента pk
        ## можем рассмотреть отдельный элемент
        model = Lessons
        template_name = 'course/lesson.html'
        context_object_name = 'lesson'



    class HomeWorksView(LoginRequiredMixin,ListView):
        model = Home_Work
        template_name = 'course/hworks.html'
        context_object_name = 'homeworks'

        def get_context_data(self,**kwargs):
            context = super().get_context_data(**kwargs)
            context['title'] = 'Домашние Задания'
            return context

    class HomeWorkView(LoginRequiredMixin,DetailView): ## класс подкласс detailview
        #чтоб рассмотреть элемент из сэта, туда пишем модель и с помощью встроенного аргумента pk
        ## можем рассмотреть отдельный элемент
        model = Home_Work
        template_name = 'course/hwork.html'
        context_object_name = 'homework'




class HomeWorkCreateView(LoginRequiredMixin,CreateView):
    model = Home_Work
    form_class = HomeWorkForm
    template_name = 'course/user_home_work.html'


    def form_valid(self, form):

        form.instance.user = self.request.user
```

Продолжение приложения Б

```
        # form.instance.slug = Text.slugify_unique(self.model, form.instance.title)
        return super().form_valid(form)


class HomeWorkUpdateView(LoginRequiredMixin,UpdateView):
    model = Home_Work
    form_class = HomeWorkForm
    template_name = 'course/user_home_work_update.html'


class HomeWorkDeleteView(LoginRequiredMixin,DeleteView):
    model = Home_Work
    success_url = reverse_lazy('HomeWorks')
    template_name = 'course/user_home_work_delete.html'



def AssignmentHome(request):
    assignment = Home_Work.objects.all()
    videos = Lessons.objects.all()
    context = {
        'homeworks': assignment,
        'videos': videos,
        'title' : 'Домашние задания',
    }
    return render(request, 'course/hworks.html', context=context)

def AssignmentVideo(request,video_id):
    assignment = Home_Work.objects.filter(video_id = video_id)
    paginator = Paginator(assignment, 20)
    page_number = request.GET.get('page')
    page_obj = paginator.get_page(page_number)
    videos = Lessons.objects.all()
    video = Lessons.objects.get(pk = video_id)

    context = {
        'homeworks' : assignment,
        'lessons' : videos,
        'lesson' : video,
        'title' : 'Домашние задания',
        'page_obj': page_obj,
    }
```

```
    return render(request,'course/hwork.html', context=context)



def user_login(request):
    if request.method == 'POST':
        form = UserLoginForm(data = request.POST)
        if form.is_valid():
            user = form.get_user()
            login(request, user)
            return redirect('Home')
    else:
            form = UserLoginForm()
    return render(request, 'course/login.html',{"form" : form, "title" : "Вход"})


def user_logout(request):
    logout(request)
    return redirect('Home')
```

6. utils.py – сервисы

```
from django.utils.text import slugify
from transliterate import translate,  detect_language

from time import time

def gen_slug(s):

    if detect_language(s):
        translate = translit(s, 'ru', reversed=True)
        new_slug = slugify(translate, allow_unicode=True)

    else:
        new_slug = slugify(s,allow_unicode=True)

    return str(new_slug) + '-' + str(int(time()))
```

7. base.html – главный шаблон всего веб-приложения

Продолжение приложения Б

```
{% load static %}
<!DOCTYPE html>
<html>
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <link rel="icon" type="image/x-icon" href="{% static 'img/logo.ico' %}">
   <link rel="stylesheet" href="{% static 'bootstrap/css/bootstrap.min.css' %}">
   <link rel="stylesheet" href="{% static 'css/main.css' %}">
   <link rel="stylesheet" href="{% static 'css/_footer.css' %}">

   <title>{% block title %}
     ASDANMUSIC
   {% endblock title %}</title>
</head>
<body>

   {% include "_nav/_nav.html" %}



   <div class="container con">

      {% block sidebar %}

      {% endblock sidebar %}

      {% block content %}

      {% endblock content %}

   </div>

   {% include "_footer/_footer.html" %}

   <script src="{% static 'bootstrap/js/bootstrap.bundle.min.js'%}"></script>
</body>
</html>
```

8. settings.py – главные настройки для правильной работы

Продолжение приложения Б

```
from pathlib import Path
import os
# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent


# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/4.0/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'django-insecure-
uk5((^cy!b3^r^wz=!3$%o8_11%y+s0v+023$figjtnk53a1@3'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []


# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'ckeditor',
    'debug_toolbar',
    'course.apps.CourseConfig',

]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
```

Продолжение приложения Б

```
'debug_toolbar.middleware.DebugToolbarMiddleware',
]

ROOT_URLCONF = 'alpha03.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [
            os.path.join(BASE_DIR, 'templates'),
        ],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]

WSGI_APPLICATION = 'alpha03.wsgi.application'


# Database
# https://docs.djangoproject.com/en/4.0/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}


# Password validation
# https://docs.djangoproject.com/en/4.0/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
```

Продолжение приложения Б

```
                'NAME':
        'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
            },
            {
                'NAME':
'django.contrib.auth.password_validation.MinimumLengthValidator',
            },
            {
                'NAME':
'django.contrib.auth.password_validation.CommonPasswordValidator',
            },
            {
                'NAME':
'django.contrib.auth.password_validation.NumericPasswordValidator',
            },
        ]


        # Internationalization
        # https://docs.djangoproject.com/en/4.0/topics/i18n/

        LANGUAGE_CODE = 'ru'

        TIME_ZONE = 'UTC'

        USE_I18N = True

        USE_TZ = True


        # Static files (CSS, JavaScript, Images)
        # https://docs.djangoproject.com/en/4.0/howto/static-files/

        STATIC_URL = 'static/'
        STATIC_ROOT = os.path.join(BASE_DIR,'static')
        STATICFILES_DIRS = [
            os.path.join(BASE_DIR, 'alpha03/static'),
        ]


        MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
        MEDIA_URL = '/media/'
```

Продолжение приложения Б

```
# Default primary key field type
# https://docs.djangoproject.com/en/4.0/ref/settings/#default-auto-field

DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'


INTERNAL_IPS = [
    '127.0.0.1',
]

CKEDITOR_UPLOAD_PATH = "uploads/"


CKEDITOR_CONFIGS = {
    'default': {
        'skin': 'moono-lisa',
        # 'skin': 'office2013',
        'toolbar_Basic': [
            ['Source', '-', 'Bold', 'Italic']
        ],
        'toolbar_YourCustomToolbarConfig': [
            {'name': 'document', 'items': ['Source', '-', 'Save', 'NewPage', 'Preview',
'Print', '-', 'Templates']},
            {'name': 'clipboard', 'items': ['Cut', 'Copy', 'Paste', 'PasteText',
'PasteFromWord', '-', 'Undo', 'Redo']},
            {'name': 'editing', 'items': ['Find', 'Replace', '-', 'SelectAll']},
            {'name': 'forms',
             'items': ['Form', 'Checkbox', 'Radio', 'TextField', 'Textarea', 'Select',
'Button', 'ImageButton',
                       'HiddenField']},
            '/',
            {'name': 'basicstyles',
             'items': ['Bold', 'Italic', 'Underline', 'Strike', 'Subscript', 'Superscript', '-',
'RemoveFormat']},
            {'name': 'paragraph',
             'items': ['NumberedList', 'BulletedList', '-', 'Outdent', 'Indent', '-',
'Blockquote', 'CreateDiv', '-',
                       'JustifyLeft', 'JustifyCenter', 'JustifyRight', 'JustifyBlock', '-',
'BidiLtr', 'BidiRtl',
                       'Language']},
            {'name': 'links', 'items': ['Link', 'Unlink', 'Anchor']},
```

Продолжение приложения Б

```
                {'name': 'insert',
                 'items': ['Image', 'Flash', 'Table', 'HorizontalRule', 'Smiley',
'SpecialChar', 'PageBreak', 'Iframe']},
                '/',
                {'name': 'styles', 'items': ['Styles', 'Format', 'Font', 'FontSize']},
                {'name': 'colors', 'items': ['TextColor', 'BGColor']},
                {'name': 'tools', 'items': ['Maximize', 'ShowBlocks']},
                {'name': 'about', 'items': ['About']},
                '/',  # put this to force next toolbar on new line
                {'name': 'yourcustomtools', 'items': [
                   # put the name of your editor.ui.addButton here
                   'Preview',
                   'Maximize',
                ]},
            ],
            'toolbar': 'YourCustomToolbarConfig',
            'tabSpaces': 4,
            'extraPlugins': ','.join([
               'uploadimage', # the upload image feature
               # your extra plugins here
               'div',
               'autolink',
               'autoembed',
               'embedsemantic',
               'autogrow',
               # 'devtools',
               'widget',
               'lineutils',
               'clipboard',
               'dialog',
               'dialogui',
               'elementspath'
            ]),
        }
    }
```