**MusicHub**

**Авторы: Полина, Кирилл**

**1. Введение**

MusicHub — веб-платформа для музыкантов и творческих людей, позволяющая публиковать работы, находить единомышленников и обмениваться контентом. Проект реализован на микрофреймворке Flask и сочетает в себе социальные функции с элементами медиа-хостинга.

Цель проекта:

Создать удобное пространство для публикации музыкального контента.

Обеспечить пользователям возможность категоризации материалов через теги.

Реализовать базовые функции социальной сети: регистрация, профиль, лента публикаций.

Целевая аудитория:

Музыканты, композиторы, звукорежиссеры.

Любители, желающие делиться своими работами.

**2. Функциональные возможности**

2.1. Аутентификация и авторизация

Регистрация в два этапа:

Ввод личных данных (email, пароль, имя пользователя).

Выбор тегов, отражающих интересы (жанры, инструменты).

Вход/выход с использованием Flask-Login.

Защита паролей через хеширование (библиотека Werkzeug).

2.2. Работа с контентом

Создание постов:

Загрузка файлов (аудио, изображения, документы).

Привязка тегов к публикациям.

Лента публикаций:

Фильтрация по тегам (например, «Рок», «Гитара»).

Поиск по заголовкам.

Пагинация (9 постов на странице).

Отображение медиа:

Аудиоплеер для MP3/WAV.

Превью изображений с возможностью увеличения.

2.3. Профиль пользователя

Загрузка аватара (JPG/PNG).

Редактирование информации "О себе".

Просмотр выбранных при регистрации тегов.

2.4. Дополнительные функции

Временная метка публикаций в формате «time ago» (например, «2 ч. назад»).

Адаптивный интерфейс с поддержкой мобильных устройств.

**3. Техническая реализация**

3.1. Архитектура

Модели данных (SQLAlchemy):

User: данные пользователя, включая аватар и био.

Post: заголовок, контент, путь к файлу, связь с автором.

Tag: жанры и инструменты.

Связующие таблицы: UserTag, PostTag.

Маршруты (Flask):

Регистрация, вход, выход.

Создание/просмотр постов, профиль.

Шаблоны: HTML + Jinja2 с использованием Bootstrap для стилизации.

3.2. Безопасность

CSRF-токены для форм (Flask-WTF).

Валидация паролей (минимальная длина — 8 символов).

Санитизация имен файлов при загрузке (secure\_filename).

3.3. Работа с файлами

Аватары сохраняются в /app/static/img/.

Файлы постов — в /app/static/uploads/.

Поддержка форматов: MP3, WAV, JPG, PNG, PDF, DOC.

3.4. Стилизация

Единый дизайн в стиле Discord:

Темная тема с акцентами (#5865F2).

Адаптивные сетки для постов и тегов.

Кастомные чекбоксы и кнопки.

**4. Использованные технологии**

Backend:

Python 3, Flask, Flask-SQLAlchemy, Flask-Login, Flask-WTF.

Frontend:

HTML5, CSS3 (Flexbox, Grid), JavaScript (базовые скрипты).

Bootstrap 5, Jinja2.

База данных:

SQLite (для разработки), миграции через Flask-Migrate.

Инструменты:

Werkzeug (безопасность), SQLAlchemy (ORM).

**5. Особенности реализации**

Динамическая фильтрация:

Запросы к БД с использованием JOIN для связи постов и тегов.

Оптимизация медиа:

Превью изображений генерируются на лету через JavaScript.

Сессии:

Данные регистрации хранятся в сессии между этапами.

Миграции:

Автоматическое создание тегов при первом запуске (см. run.py).

**6. Как можно продолжить разворачивать данный проект:**

Безопасность:

Добавить подтверждение email.

Внедрить ограничение скорости запросов (Flask-Limiter).

Функционал:

Комментарии и лайки к постам.

Система подписок.

Оптимизация:

Кэширование через Redis.

Использование CDN для статики.

Масштабирование:

Переход на PostgreSQL.

Разделение на микросервисы (например, отдельный сервис для загрузки файлов).

**7. Заключение**

Проект MusicHub демонстрирует возможности Flask для создания полноценных веб-приложений. Реализованы ключевые функции социальной сети с упором на музыкальный контент. Кодовая база структурирована, что упрощает дальнейшее развитие. Текущая версия уже готова для тестирования и использования в небольших сообществах.