ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ

«Университет «Дубна»

ИНСТИТУТ СИСТЕМНОГО АНАЛИЗА И УПРАВЛЕНИЯ

**Отчёт**

по дисциплине

**«Разработка приложений баз данных»**

«Разработка приложения «Сервис текстов песен»»

**ТЕМА**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наименование темы)

3012 Тютявин Никита Владимирович

**Выполнил**: студент группы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(Ф.И.О.)

**Руководитель**:

Иванцова О.В.

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Дубна, 2024

Содержание

[Введение 3](#_Toc185951691)

[Постановка задачи 4](#_Toc185951692)

[Реализация базы данных 5](#_Toc185951693)

[Реализация интерфейса приложения 11](#_Toc185951694)

[Авторизация пользователя 11](#_Toc185951695)

[Личный кабинет (пользователь без роли модератора) 13](#_Toc185951696)

[Главная страница 16](#_Toc185951697)

[Страница песни 17](#_Toc185951698)

[Просмотр карточки артиста 19](#_Toc185951699)

[Поиск песен 20](#_Toc185951700)

[Возможности пользователя с ролью модератора 20](#_Toc185951701)

[Заключение 24](#_Toc185951702)

# Введение

В современном цифровом мире растет потребность в интерактивных сервисах, которые облегчают доступ к информации и способствуют обмену мнениями. Одним из таких востребованных направлений является создание платформ для взаимодействия пользователей на основе музыкального контента. Введение приложения «Сервис текстов песен» отвечает этой потребности, предоставляя пользователям возможность не только искать и просматривать тексты песен, но и активно участвовать в обсуждении их содержания.

В данном проекте будет приведена документация по реализации веб-приложения «Сервис текстов песен».

# Постановка задачи

**Цель работы**: создать интерактивный сервис для поиска, просмотра и добавления текстов песен, а также для взаимодействия пользователей через комментарии и объяснения к текстам.

**Исходный данные:** схема БД «Сервис текстов песен», документация к средствам реализации задачи.

**Априорные представления**: веб-приложение, которое может реализовать следующие функции:

1. Пользователь. Регистрация
2. Пользователь. Авторизация
3. Пользователь. Изменение данных аккаунта
4. Пользователь. Поиск песен по названию, исполнителю и тексту песни
5. Пользователь. Просмотр информации и слов песни
6. Пользователь. Возможность добавлять новую песню и строки текста (требуется проверка модератором). К каждой строке (или сразу к нескольким) может быть оставлено объяснение слов
7. Пользователь. Оценить объяснение слов, добавить к нему комментарий
8. Пользователь. Оставить комментарий к песне
9. Модератор. Принять песню
10. Модератор. Принять объяснение к показу
11. Модератор. Удалить песню
12. Модератор. Изменить песню
13. Модератор. Изменить данные артиста
14. Модератор. Удалить артиста
15. Пользователь. Просмотр и выставление рейтинга артисту и песням артистов

**Средства реализации:** СУБД PostgreSQL, JavaScript (фреймворк Vue.js)

**Ожидаемый результат:** готовое веб-приложение, которое реализует все перечисленные выше функции.

**Критерии оценки:**

* Приложение с успехом проходит тесты на устойчивость системы и корректное функционирование всех функций.

# Реализация базы данных

В приложение были создан следующий набором основных таблиц для работы (см. Рисунок 1).

Изображение выглядит как снимок экрана, текст

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1. Логическая схема БД

**Исходные данные:**

Таблица *users* – данные о всех пользователях, зарегистрированных в приложении

* *Id (Primary Key) int:* уникальный идентификатор пользователя.
* *Username (UNIQUE)* text: имя пользователя.
* *Email (UNIQUE)* text: почта пользователя.
* *Password text*: пароль от пользовательского аккаунта.
* *Role bool*: роль пользователя (*False* – обычный пользователь, *True* - модератор)
* *Profile\_icon\_path text*: ссылка на нынешнюю аватарку пользователя

Таблица *artists* – данные о всех добавленных артистах

* *Id (Primary Key) int*: уникальный идентификатор артиста.
* *Name text*: псевдоним артиста.
* *Real\_name text*: настоящее имя пользователя.
* *description text*: описание артиста.

Таблица *songs* – данные о всех добавленных песнях

* *Id (Primary Key) int*: уникальный идентификатор песни.
* *Title text*: название песни.
* *Artist (Foreign Key from artists) int*: уникальный идентификатор артиста.
* *Added\_by (Foreign Key from users) int*: уникальный идентификатор пользователя, добавившего песню.
* *Is\_approved bool*: значение, определяющее, подтверждена песня или нет.
* *Search\_vector tsvector*: атрибут, используемый для реализации полнотекстового поиска в приложении.
* *Created\_at timestamptz*: время создания песни (с указанием часового пояса).
* *Description text*: описание песни.

Таблица *artists\_ratings* – данные о рейтингах артистов

* *Artist\_id (Primary Key, Foreign Key from artists) int*: уникальный идентификатор артиста.
* *Added\_by (Primary Key, Foreign Key from users) int*: уникальный идентификатор пользователя, добавившего песню.
* *Rating int*: рейтинг, поставленный пользователем (присваивается 1, если рейтинг был поставлен пользователем).

Таблица *songs\_lyrics* – тексты песен

* *Id (Primary Key, Foreign Key from songs) int*: уникальный идентификатор песни.
* *Lyrics\_text text*: текст песни.

Таблица *lyrics\_explanations* – объяснения, добавленные пользователями к песням

* *Id (Primary Key) int*: уникальный идентификатор объяснения.
* *Lyric\_id (Foreign Key from songs\_lyrics) int*: уникальный идентификатор текста песни.
* *Explanation text*: текст объяснения.
* *Idx\_start int*: индекс начала объяснения в тексте.
* *Idx\_end int*: индекс конца объяснения в тексте.
* *Is\_approved bool*: подтверждено ли объяснение администратором.
* *Added\_by (Foreign Key from users) int*: уникальный идентификатор пользователя, добавившего объяснение.

Таблица *songs\_ratings* – хранит рейтинги пользователей, оставленные к песням

* *Song\_id (Primary Key, Foreign Key from songs) int*: уникальный идентификатор песни.
* *Added\_by (Primary Key, Foreign Key from users) int*: уникальный идентификатор пользователя, добавившего объяснение.
* *Rating int*: рейтинг, поставленный пользователем (присваивается 1, если рейтинг был поставлен пользователем).

Таблица *explanations\_ratings* – рейтинги объяснений к песням

* *Explanation\_id (Primary Key, Foreign Key from explanations) int*: уникальный идентификатор объяснения.
* *Added\_by (Primary Key, Foreign Key from users) int*: уникальный идентификатор пользователя, добавившего объяснение.
* *Rating int*: рейтинг, поставленный пользователем (присваивается 1, если была поставлена положительная оценка, -1 если была поставлена отрицательная оценка).

Таблица song\_comments – комментарии пользователей к песням

* *id (Primary Key) int*: уникальный идентификатор комментария.
* *Created\_at timestamptz*: время, в которое был добавлен комментарий с указанием часового пояса.
* *Song\_id (Foreign Key from songs) int*: уникальный идентификатор песни.
* *Added\_by (Foreign Key from users) int*: уникальный идентификатор пользователя.
* *Text text*: текст комментария.

Таблица *explanations\_comments* – комментарии пользователей к объяснениям

* *id (Primary Key) int:* уникальный идентификатор комментария.
* *Created\_at timestamptz*: время, в которое был добавлен комментарий с указанием часового пояса.
* *Explanation\_id (Foreign Key from lyrics\_explanations) int*: уникальный идентификатор объяснения.
* *Added\_by (Foreign Key from users) int*: уникальный идентификатор пользователя.
* comment text: текст комментария.

Также для удобства работы с данными были созданы 3 материализованных представления: *artists\_ratings\_aggr* – содержит агрегацию рейтинга артистов, *songs\_ratings\_aggr* – содержит агрегацию рейтинга песен, *explanations\_ratings\_aggr* – агрегация рейтингов объяснений. Реализации материализованного представления показана на примере *artists\_ratings\_aggr* (см. Рисунок 2).

Изображение выглядит как текст, Шрифт, линия, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2. Пример материализованного представления artists\_ratings\_aggr

Для реализации работы приложения были созданы следующие функции:

* *register\_userupd* (*email text, password text, username text*) – заносит информацию о новом пользователе в базу данных.
* authenticate\_user (email\_or\_login text, password text) – возвращает данные о пользователе при успешной авторизации
* update\_search\_vector () – триггерная функция, обновляющая значение *search\_query* в таблице *songs,* привязанная к триггеру уровня записи *trigger\_update\_search\_vector,* реагирующий после *INSERT* и *UPDATE* на таблице *songs* (добавляет возможность поиска по имени артиста и названию песни)
* *update\_lyrics\_search\_vector* () – триггерная функция, обновляющая значение *search\_query* в таблице *songs,* привязанная к триггеру уровня записи *trigger\_update\_lyrics\_search\_vector,* реагирующий после *INSERT* и *UPDATE* на таблице *songs\_lyrics* (добавляет возможность поиска по словам песни)
* *search\_songs (search\_query text) –* возвращает песню по введенному запросу
* *fetch\_lyrics (song\_id int8) –* возвращает данные песни по ее идентификатору
* *add\_explanation (lyric\_id int8, explanation text, idx\_start int8, idx\_end int8, added\_by int8) –* функция добавления объяснения в базу данных
* *fetch\_explanations (lyric\_id int8) –* возвращает все утвержденные объяснения к песне
* *add\_users\_profile\_icon (user\_id int8, path text) –* добавляет путь к иконке профиля, сохраненной на базе данных
* *fetch\_users\_songs (user\_id int8) –* возвращает все песни, добавленные пользователем
* *delete\_user\_song (p\_song\_id int8) –* каскадное удаление данных о песне из всех таблиц
* *function fetch\_artists() –* возвращает всех добавленных артистов
* *add\_artist (p\_name text, p\_real\_name text, p\_description text) –* добавляет артиста в базу
* *add\_song (p\_name text, p\_lyrics text, p\_description text, artist\_id int8, user\_id int8) –* добавляет песню в базу
* *fetch\_most\_rated\_artists () –* возвращает 10 артистов с самыми высокими рейтингами
* *fetch\_most\_rated\_songs () –* возвращает 10 артистов с самыми высокими рейтингами
* *get\_songs\_added\_today () –* возвращает все песни, добавленные сегодня
* *fetch\_artist\_info (artist\_id int8) –* возвращает информацию о конкретном артисте
* *refresh\_artists\_ratings () –* триггерная функция, которая обновляет материализованное представление *artists\_ratings\_aggr,* привязана к строковому триггеру *refresh\_artists\_ratings\_trigger,* срабатывающемупосле *INSERT*, *UPDATE*, *DELETE* на таблице *artists\_ratings*
* *refresh\_songs\_ratings () –* триггерная функция, которая обновляет материализованное представление *songs\_ratings\_aggr,* привязана к строковому триггеру *refresh\_songs\_ratings\_trigger,* срабатывающему после *INSERT*, *UPDATE*, *DELETE* на таблице *songs\_ratings*
* *fetch\_explanation\_rating (p\_explanation\_id int8) –* возвращает агрегированный рейтинг объяснения
* *add\_rating\_to\_explanation (p\_explanation\_id int8, p\_new\_rating int8, p\_user\_id int8) –* добавляет рейтинг к объяснению от пользователя в базу
* *refresh\_explanations\_ratings () –* триггерная функция, обновляет материализованное представление *explanations\_ratings\_aggr,* привязана к триггеру уровня записи *refresh\_explanations\_ratings\_trigger,* срабатывающему после *INSERT*, *UPDATE*, *DELETE* на таблице *explanations\_ratings*
* *fetch\_artist\_info (artist\_id int8) –* возвращает данные о конкретном артисте
* *fetch\_artist\_songs (artist\_id int8) –* возвращает все песни артиста
* *add\_comment(p\_song\_id int8, p\_user\_id int8, p\_text text) –* функция добавления комментария к песне
* *fetch\_comments (p\_song\_id int8) –* возвращает комментарии к песне
* *add\_explanation\_comment (p\_user\_id int8, p\_comment text, p\_expl\_id int8) –* добавляет комментарий пользователя к объяснению
* *fetch\_explanation\_comments (p\_expl\_id int8) –* возвращает все комментарии к объяснению
* *add\_rating\_to\_song (p\_song\_id int8, p\_user\_id int8) –* добавляет рейтинг к песне
* *fetch\_song\_rating (p\_song\_id int8) –* возвращает рейтинг песни
* *add\_rating\_to\_artist (p\_artist\_id int8, p\_user\_id int8) –* добавляет рейтинг артисту
* *fetch\_artist\_rating (p\_artist\_id int8) –* возвращает рейтинг артиста
* *delete\_song\_comment (p\_comment\_id int8) –* удаляет комментарий к песне
* *delete\_explanation\_comment (p\_expl\_comment\_id int8) –* удаляет комментарий к объяснению
* *fetch\_unapproved\_songs () –* возвращает все неподтвержденные песни
* *approve\_explanation (p\_explanation\_id int8)* – делает объяснение подтвержденным (позволяет показывать объяснение песни)
* *delete\_explanation (p\_explanation\_id int8) –* удаляет объяснение
* *fetch\_unapproved\_explanations () –* возвращает неподтвержденные объяснения
* *approve\_song (p\_song\_id int8) –* принимает песню к показу в приложении
* *update\_song\_data (p\_song\_id int8, p\_lyrics text default null, p\_description text default null, p\_song\_name text default null) –* обновляет данные песни
* *update\_user (p\_user\_id int8, new\_username text default null, new\_email text default null, new\_password text default null) –* обновляет данные пользователя
* *delete\_user (p\_user\_id int8) –* каскадное удаление пользователя
* *update\_artist (p\_artist\_id int8, new\_name text default null, new\_real\_name text default null, new\_description default null) –* функция обновления данных об артисте
* *delete\_artist (p\_artist\_id int8) –* каскадное удаление артиста

Рассмотрим работу функции на примере *fetch\_artist\_rating* (см. Рисунок 3). В данной функции мы ищем рейтинг по *id* артиста в материализованном представлении, после чего возвращаем его для дальнейшей обработки в приложение, также добавляем характеристику *security definer,* чтобы выдать пользователю, запускающему функцию права пользователя, владеющего этой функцией.

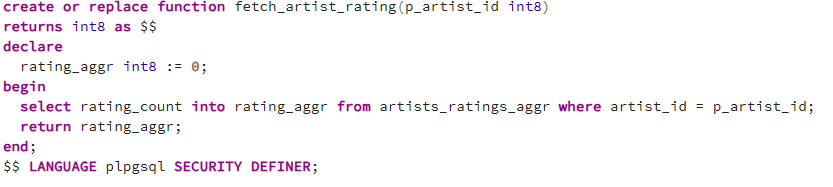


Рисунок 3. Пример функции fetch\_artist\_rating

Теперь рассмотрим функцию регистрации – в качестве входного параметра эта функция получает *email*, пароль и имя пользователя, в случае успешного добавления ничего не возвращает, а в случае, если такой пользователь уже существует, возвращает пользователю ошибку об этом(см. Рисунок 4).

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4. Пример работы процедуры register\_userupd

# Реализация интерфейса приложения

## Авторизация пользователя

Пользователь может авторизоваться и зарегистрироваться на странице */auth*. При успешной регистрации или авторизации, все функции приложения станут доступны (в зависимости от роли пользователя) (см. Рисунок 5). Также доступен переход на главную страницу приложения (см. Рисунок 6).

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 5. Формы авторизации и регистрации

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 6. Главная страница для неавторизованного пользователя

При попытке авторизации с некорректными данными будет выброшено предупреждение (см. Рисунок 7). При попытке регистрации с данными, присутствующими в таблице, будет выброшено сообщение об ошибке (см. Рисунок 8).

Изображение выглядит как текст, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 7. Обработка ввода некорректных данных при авторизации

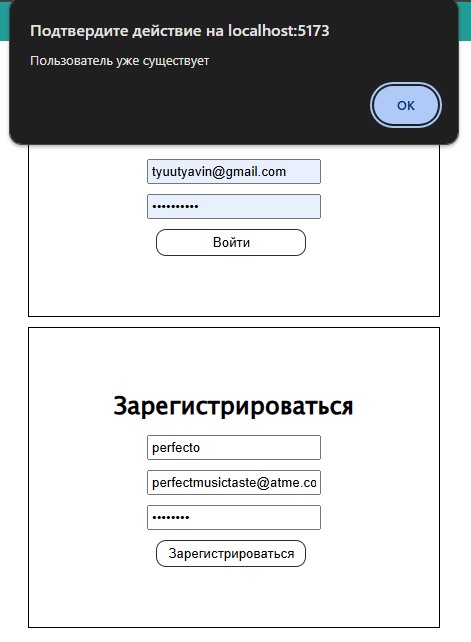


Рисунок 8. Обработка ввода некорректных данных при регистрации

## Личный кабинет (пользователь без роли модератора)

После обработки правильных данных пользователь сможет перейти на страницу профиля (см. Рисунок 9). В зависимости от роли ему может быть доступна функция “Перейти на страницу контрибьютора” для работы с неподтвержденными песнями и объяснениями. Так как, помимо этого, у пользователя с ролью модератора появляется много новых функций, рассмотрим их все позже. На каждой страничке авторизованного пользователя доступна кнопка выхода, меню, кнопка перехода в профиль и поиск песен (для неавторизованного доступно все, кроме перехода в профиль и кнопки выхода, вместо этого он может нажать на кнопку “Вход/Регистрация”).

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 9. Личный кабинет обычного менеджера

Пользователь может добавить песню нажатием кнопки, которая выводит форму добавления песни, с возможностью добавить название песни, автора песни, текст песни, описание. Если артист не был добавлен ранее, пользователь может добавить артиста, введя его псевдоним, настоящее имя и его описание (см. Рисунок 10). После введения всех данных, песня добавляется в профиль как неподтвержденная, ее можно удалить (см. Рисунок 11).

Изображение выглядит как текст, электроника, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 10. Форма добавления песни

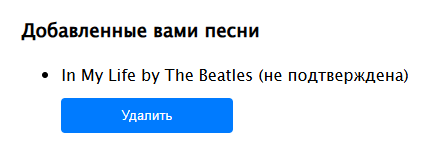


Рисунок 11. Отображение добавленной неподтвержденной песни

Если песня была подтверждена модератором, то у пользователя отобразится ссылка на песню, с возможностью удалить ее (см. Рисунок 12)

Изображение выглядит как текст, снимок экрана, Шрифт, Цвет электрик

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 12. Отображение добавленной подтвержденной песни

Также, пользователь может добавить иконку профиля, выбрав изображение и отправив его на базу, нажав на кнопку “Загрузить картинку”.

Пользователь может изменить аккаунт, нажав на кнопку “Изменить аккаунт”, где он сможет поменять пароль, имя и почту (см. Рисунок 13).

Изображение выглядит как текст, снимок экрана, число, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 13. Форма изменения аккаунта

## Главная страница

При нажатии на логотип “*lyrical*” пользователь переходит на главную страницу приложения, где отображаются рейтинги артистов, песен, и новых добавленных песен (см. Рисунок 14). Пользователь может ставить лайки любимым артистам и песням, убирать лайки, обновление страницы для этого не требуется. Также, можно перейти на страницу любого артиста, песни и посмотреть подробную информацию.

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 14. Главная страница

## Страница песни

На страницу песни можно попасть через поиск, найдя любую интересующую песню, присутствующую в приложении, через главную страницу посмотреть популярную песню, или через профиль перейти на добавленную вами песню, если она была подтверждена. На ней расположена информация о песне, ссылка на автора песни, текст песни, кнопка лайка, внизу расположены комментарии и форма добавления комментария (см. Рисунок 15, 16). Неавторизованный пользователь не может поставить лайк или добавить комментарий.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 15. Страница текста песни

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 16. Комментарии на странице текста песни

Пользователь может добавить объяснение к песне, выделив необходимые строчки/слова, после чего нажав кнопку добавить объяснение и добавить текст объяснения, нажать кнопку сохранить и объяснение уйдет на подтверждение модератору (см. Рисунок 17). После утверждения объяснение будет показываться у пользователя (необходимо обновление страницы) (см. Рисунок 18). К подтвержденному объяснению можно добавить комментарий и оценить его положительно или отрицательно. Пользователь, добавивший комментарий, может удалить его, а также убрать поставленный рейтинг нажатием по цифре рейтинга (см. Рисунок 19). Также, объяснение можно удалить, если это объяснение добавил пользователь, авторизованный в системе в данный момент.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 17. Добавление объяснения к словам

Изображение выглядит как текст, Шрифт, снимок экрана, веб-страница

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 18. Отображение подтвержденного объяснения

Изображение выглядит как текст, снимок экрана, веб-страница, Веб-сайт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 19. Объяснение с добавленным комментарием

## Просмотр карточки артиста

Пользователь может перейти на страницу артиста, нажав на ссылку на главной странице (в рейтинге артистов) или с помощью страницы текста песни. Обычный пользователь может поставить лайк артисту, убрать лайк, посмотреть описание артиста, посмотреть все песни артиста (см. Рисунок 20).

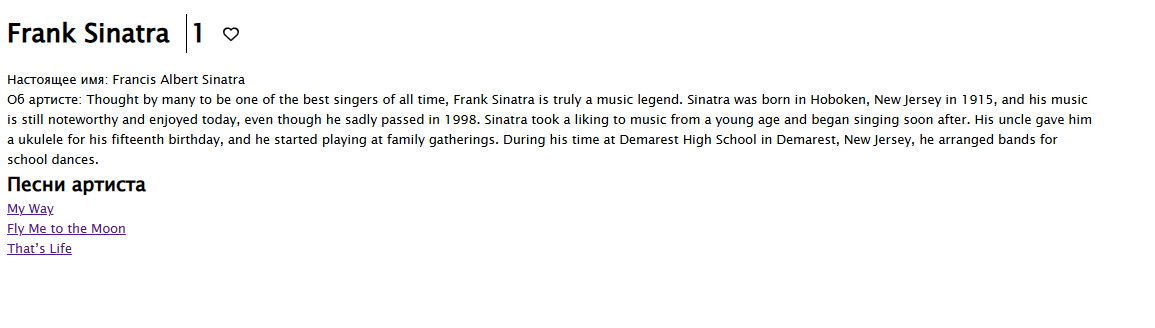


Рисунок 20. Карточка артиста

## Поиск песен

Пользователь может найти песню по названию, имени артиста или по тексту песни

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 21. Поиск песен

## Возможности пользователя с ролью модератора

Модератор может перейти на страницу профиля и перейти на страницу контрибьютора (см. Рисунок 22).

Изображение выглядит как текст, снимок экрана, логотип, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 22. Профиль с ссылкой на страницу модератора

На данной странице модератор может посмотреть неподтвержденные песни и их текст, подтвердить их или удалить, посмотреть непринятые объяснения, посмотреть к каким словам они оставлены, подтвердить объяснение или удалить его (см. Рисунок 23).

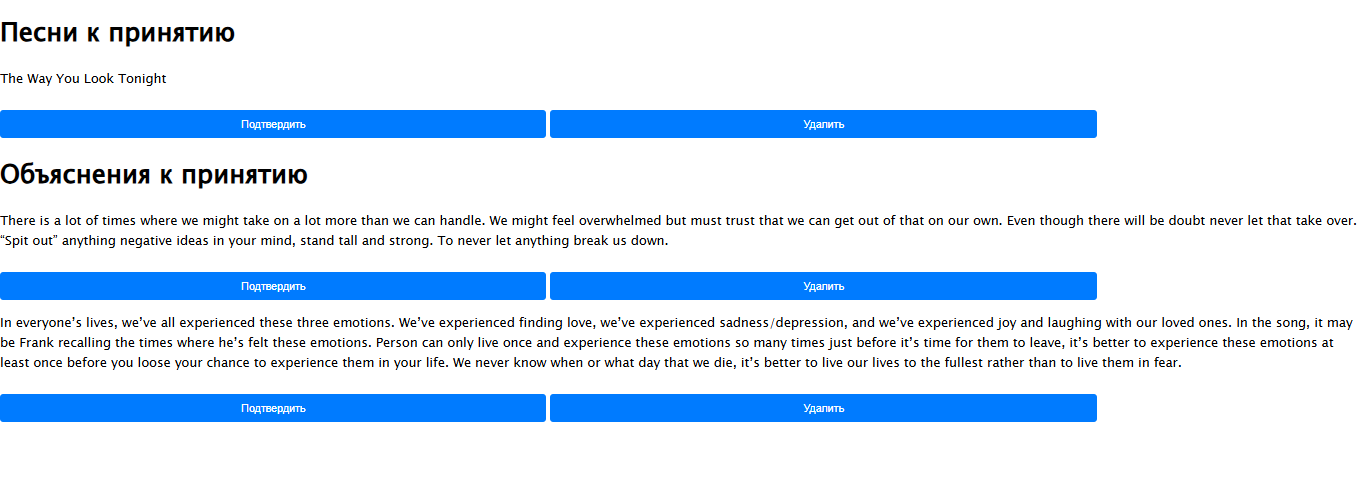


Рисунок 24. Страница модератора

Также, на каждой странице песни или артиста, модератор может обновить всю информацию об артисте или песне, в зависимости от того, на какой странице он находится, а также удалить артиста или песню (см. Рисунок 25, 26).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 25. Изменение и удаление песни

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 26. Изменение и удаление артиста

Модератор также может удалить объяснение, добавленное любым пользователем нажатием кнопки “Удалить” в выбранном объяснении (см. Рисунок 27).

Изображение выглядит как текст, снимок экрана, веб-страница, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 27. Кнопка удаления объяснения

# Заключение

Веб-приложение «Сервис текстов песен» было успешно разработано для пользователей, обеспечивая все заранее заданные функции. Это приложение способствует быстрому и удобному поиску, просмотру и добавлению текстов песен, а также взаимодействию пользователей через комментарии и объяснения благодаря интуитивному интерфейсу и тщательной проверке данных. Приложение удовлетворяет всем априорным представлениям.