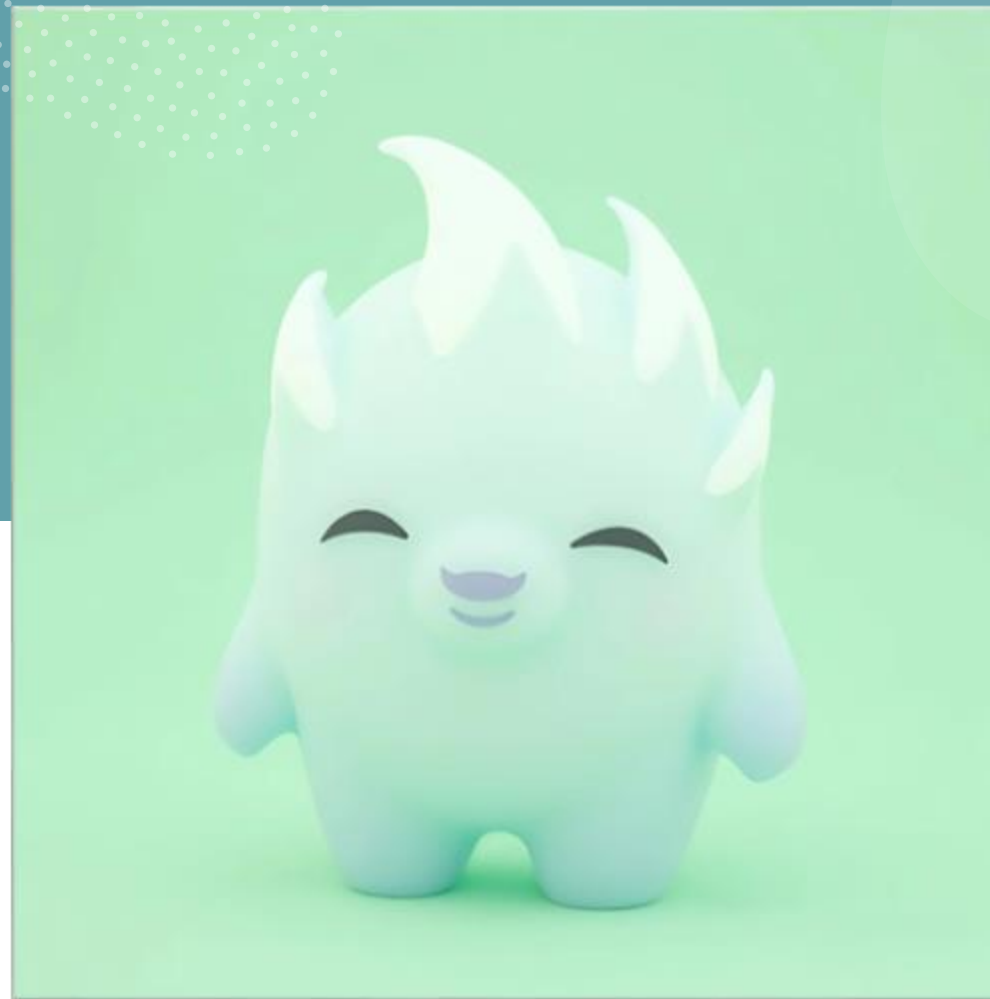


Сервис для
загрузки музыки
YaMaBot




Порой люди хотят слушать музыку, но с этим возникают проблемы. Например нельзя бесплатно прослушать свой любимый плейлист или на закончился интернет.

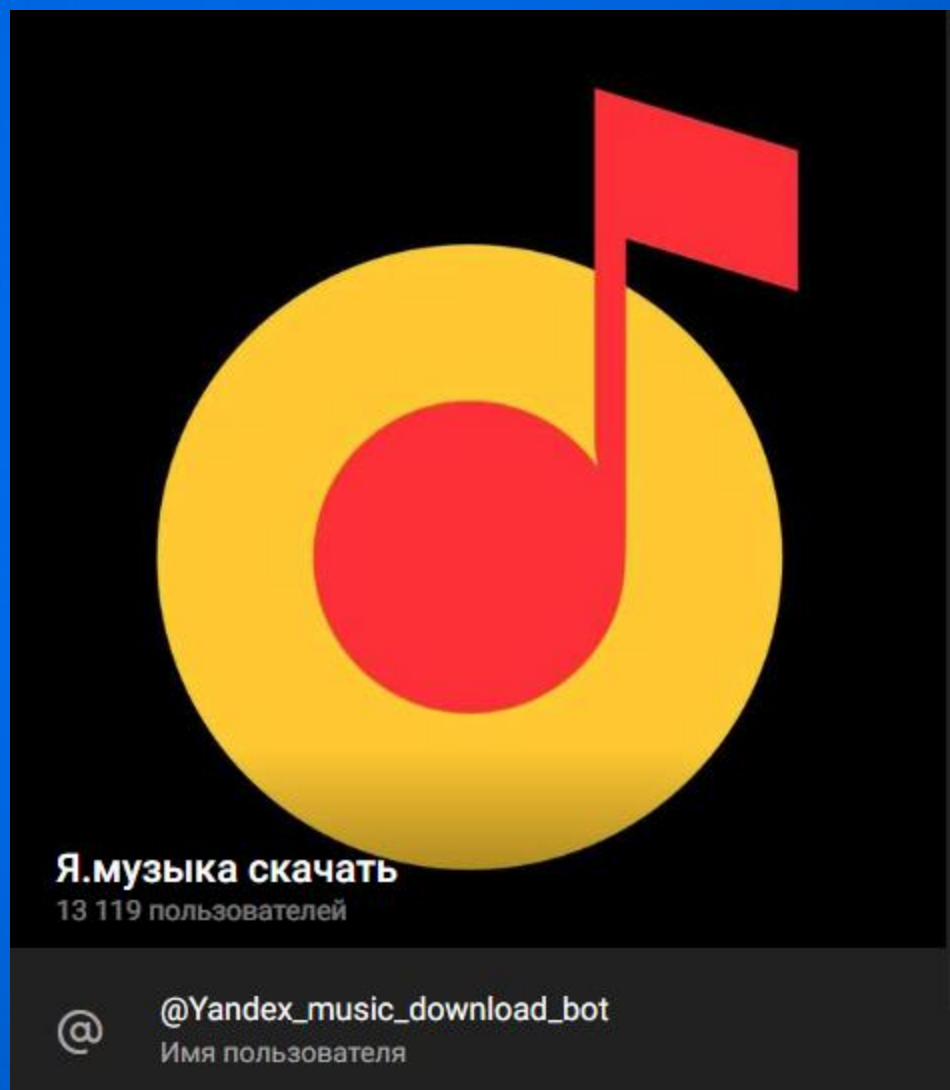


Цель



Разработать сайт и бота в Telegram для
бесплатной загрузки и прослушивания
музыки.





Перестал работать

Технологический стек



Python



Git



SQLite



Яндекс Музыка



PyCharm



Flask/
Flask-RESTful

Основные функции

```
@router.message()  # LaOgree *
async def search_tracks(message: Message):
    user_input = message.text.strip()
    if not user_input:
        await message.answer("Пожалуйста, введите название трека.")
        return

    search_msg = await message.answer("Ищу похожие треки...")
    try:
        response = requests.get(API_URL_SIMILAR + user_input)
        if response.status_code != 200:
            await search_msg.delete()
            await message.answer("Произошла ошибка при поиске трека.")
            return
        data = response.json()
        if "error" in data or not data.get("similar_tracks"):
            await search_msg.delete()
            await message.answer("Я не смог найти подходящий трек.")
            return
        tracks = data["similar_tracks"]
        keyboard = InlineKeyboardMarkup(
            inline_keyboard=[
                [
                    InlineKeyboardButton(
                        text=f"{track['title']} - {' '.join(track['artists'])}",
                        callback_data=f"track_{track['id']}"
                    )
                ]
                for track in tracks
            ]
        )
        await search_msg.delete()
        await message.answer(text="Выбери трек из предложенных вариантов:", reply_markup=keyboard)
    except Exception as e:
        await search_msg.delete()
        await message.answer("Произошла ошибка при поиске трека.")
```

```
@app.route(rule: '/find_album', methods=['GET', 'POST'])  # NazarEE127
def find_album():
    return render_template("find_album.html")

@app.route(rule: '/download_track_api/<string:track>', methods=['GET', 'POST'])  # NazarEE127
def download_track_api(track):
    req = f"http://127.0.0.1:5000/api/v1/track/{track}"
    response = requests.get(req).json()
    filename = os.path.basename(response["filename"])
    file_path = os.path.join '..', 'temp', filename
    abs_file_path = os.path.abspath(file_path)
    if not os.path.exists(abs_file_path):
        print(f"Файл не найден: {abs_file_path}")
        abort(code=404, description="Файл не найден")
    return send_file(abs_file_path, as_attachment=True)

@app.route(rule: '/download_track', methods=['GET', 'POST'])  # NazarEE127
def download_track():
    return render_template("download_track.html")

@app.route(rule: '/find_album_api/<string:album>', methods=['GET', 'POST'])  # NazarEE127
def find_album_api(album):
    req = f"http://127.0.0.1:5000/api/v1/album/{album}"
    response = requests.get(req).json()
    return response
```

Архитектура базы данных

```
class User(db.Model, UserMixin): 5 usages  NazarEE127
    id = db.Column(db.Integer, primary_key=True)
    email = db.Column(db.String(50), unique=True)
    password = db.Column(db.String(50), nullable=False)
    ava = db.Column(db.String, nullable=False)
```

	Имя	Тип данных	Первичный ключ	Внешний ключ	Уникальность	Проверка	Не NULL	Сравнение	Сгенерированное	
1	id	INTEGER	🔑				🚫			NULL
2	email	VARCHAR (50)			🔑					NULL
3	password	VARCHAR (50)					🚫			NULL
4	ava	VARCHAR					🚫			NULL

```
DB_PATH = os.path.join("bot\\db\\base.db")
base = sqlite3.connect(DB_PATH, check_same_thread=False)
cur = base.cursor()

cur.execute("""
    CREATE TABLE IF NOT EXISTS users (
        id_user INTEGER PRIMARY KEY,
        name TEXT
    )
""")

cur.execute("""
    CREATE TABLE IF NOT EXISTS users_tg (
        id_user INTEGER PRIMARY KEY
    )
""")

base.commit()
```

	Имя	Тип данных	Первичный ключ	Внешний ключ	Уникальность	Проверка	Не NULL	Сравнение	Сгенерированное	
1	id_user	INTEGER	🔑							NULL
2	name	TEXT								NULL

Плюсы и минусы проекта

Плюсы:

- Бесплатное использование
- Понятный интерфейс

Минусы:

- Малое количество функций

Перспективы

- 1) Добавить рекомендацию музыки с помощью ИИ
- 2) Добавить возможность создания своих плейлистов