



Univerzitet u Zenici
Politehnički fakultet
2024



Web dizajn

Prva/druga godina I. ciklus

Dokumentacija projekta

-MUSIC STREAMING SERVICE-

Članovi/ce tima:

- Safet Imamović
- Eman Palavra
- Dawud Žigo

Table of contents

Intro	2
Pregled Uputa i Pravila za Izradu Projekta	3
Opis	5
Raspoređivanje Zadataka	6
Baza Podataka	7
Supabase Postavke	8
Integracija Supabase-a s TypeScript Tipovima	15
Postavljanje Supabase Zavisnosti	19
Eksterna HTML Stranica za Kontakt - Slanje na Bazu	23
Inicijalno Postavljanje Layouta	26
Napomene	32

Intro


Detalji Projekta

Ime Teme: **Music Streaming Service**

Univerzitet	Fakultet	Predmet	Godina	Ciklus
Univerzitet u Zenici	Politehnički fakultet	Web Dizajn	Prva/Druga	Prvi

Članovi Tima
Safet Imamović
Eman Palavra
Dawud Žigo

Pregled Uputa i Pravila za Izradu Projekta

 Rok za predaju: 03.05.2024. do 06:00

- Projekti se rade u grupama od po isključivo 3 člana.
- Mogu se miješati studenti iz više grupa.
- Isključivo iskoristiti obrazac za dokumentaciju koji je priložen.
- Projekti koji nemaju priloženu dokumentaciju ili nisu spremljeni i postavljeni prema uputama neće se uzeti u razmatranje.
- Rad članova grupe će se ocijeniti pojedinačno (neće članovi nužno imati istu ocjenu).
- Projekat nosi 30% ukupne ocjene predmeta.
- Projekat je obavezan za polaganje predmeta.
- Tema i razvojni okvir su proizvoljni, samo je važno imati sve elemente koji su navedeni kao zadaci ovog projekta.
- Ocjenjuju se: Kvalitet i preglednost koda, kvalitet dokumentacije, inovativnost i kreativnost (web DIZAJN!) te rad prema pravilima.

Zadaci

Napraviti dinamičku web stranicu gdje ćete obavezno koristiti neki razvojni okvir za razvoj web stranica po želji sa sljedećim komponentama:

- Početna stranica ili landing page sa menijem.
- Meni treba imati opcije “o nama”, “prijava” i “kontakt”, a ostale opcije postavite prema temi projekta i po želji.
- “Prijava” treba omogućiti registraciju ili prijavu korisnika.

- Korisnici trebaju imati ulogu "Admin" ili "Guest".
- Razlike u privilegijama Admin i Guest korisnika primijenite po želji.
- Tri forme trebaju postojati (prijava, registracija i kontakt).
- Podaci koji se unose preko formi trebaju biti poslani na neku bazu podataka ili JSON datoteku po želji.
- U sklopu kontakt stranice uvrstiti Google Maps.

Bodovanje

- Pridržavanje pravila i kvalitetna dokumentacija: 10 bodova.
- Kvalitet koda i prisutnost svih obaveznih elemenata navedenih gore: 15 bodova.
- Kreativnost: 5 bodova.



NAPOMENA: Teme se moraju razlikovati. Grupa koja prva predloži neku temu, tema je njihova.



Projekat (sve fajlove i eventualno skripte za bazu i dokumentaciju) spakovati u .zip ili .rar i postaviti na zadatak na Classroom. Projekat imenovati na sljedeći način prema rednom broju vaše grupe sa liste grupa:

redniBrojGrupe_brIndeksaPredstavnik.zip ili

redniBrojGrupe_brIndeksaPredstavnik.rar

Naše: 14_MusicStreamingService_332.rar

Opis

Tema projekta je **Music Streaming Service**. Cilj projekta je kreirati web aplikaciju za strimovanje muzike koja koristi Supabase kao backend servis. Aplikacija će omogućiti korisnicima da slušaju muziku, prave plejliste, prate omiljene izvođače i albume, kao i da pretražuju muziku po različitim kriterijumima.

Kao glavne funkcionalnosti, aplikacija će imati:

- Registraciju i prijavu korisnika
- Pretragu muzike po različitim kriterijumima
- Kao Administrator: dodavanje muzike
- Lajkovanje muzike
- Kontakt (sa google maps integracijom) i O nama stranice, slanje input form detalja na bazu

Raspoređivanje Zadataka

Safet Imamović	Eman Palavra	Dawud Žigo
Baza Podataka: <ul style="list-style-type: none"> • PostgreSQL • SupaBase 	Početna stranica sa menijem	Query pjesama i prikaz liste
Kontakt stranica INSERT I NTO bazu	Implementacija Formi: <ul style="list-style-type: none"> • O Nama • Kontakt 	Funkcionalnost lajkanja pjesama
Implementacija uloga: <ul style="list-style-type: none"> • Admin: Upload pjesama sa slikama • Guest 	Main layout dizajn	Player funkcionalnost
Prijava/Login stranica	Integracija Google Maps na Kontakt stranicu	

Baza Podataka

Pregled Supabase-a

1. Šta je Supabase?

- Supabase je open-source alternativa za Firebase.
- Koristi PostgreSQL za svoju bazu podataka.
- Uključuje funkcije poput skladištenja, autentifikacije i real-time mogućnosti.
- Podržava slanje emailova za resetovanje lozinke sa prilagodljivim email šablonima.

Supabase Postavke

Kreiranje Računa

1. Kreiranje Supabase računa putem GitHub-a

- Registracija na Supabase koristeći GitHub račun.


Postavke Projekta

1. Postavke projekta

- Naziv projekta: `MusicStreamingService`
- Region: Frankfurt, EU (lokacija servera)

2. Lozinka projekta

- Generisana je lozinka za projekat i kopirana je u `password.txt` u korijenu projekta.

-  Dodano je `password.txt` u `.gitignore` kako bi se osiguralo da se ne pošalje u repozitorij.

API Ključevi

1. Postavke API ključeva

- U postavkama Supabase -> API.
- Prikupljeno je sljedeće:
 - `NEXT_PUBLIC_SUPABASE_URL`
 - `NEXT_PUBLIC_SUPABASE_ANON_KEY`
 - `SUPABASE_SERVICE_ROLE_KEY`

- Dodano je ovo u `.env.local` datoteku.
- Osigurajte da je `.env.local` dodan u `.gitignore`.

⚠ Napomena Objašnjenje API ključeva:

- `NEXT_PUBLIC_SUPABASE_URL`: URL vaše Supabase instance.
- `NEXT_PUBLIC_SUPABASE_ANON_KEY`: Anonimni ključ za javni pristup.
- `SUPABASE_SERVICE_ROLE_KEY`: Ključ sa povišenim ovlaštenjima za server-side operacije.

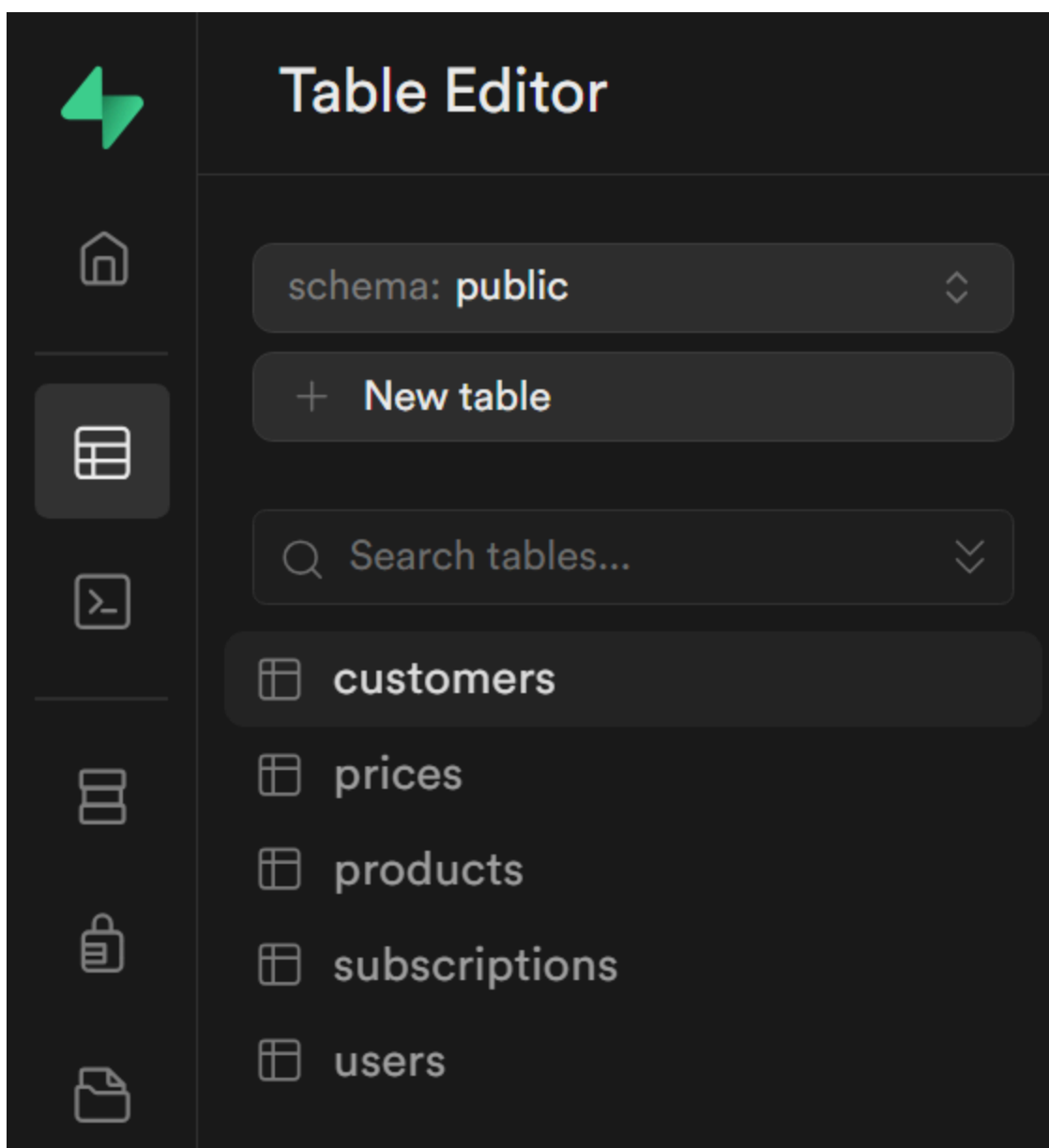
Šema Baze Podataka

Brzi Start Šabloni

1. Stripe Pretplate Šablon

Link za Stripe Pretplate Šablon

(<https://github.com/SafetImamovic/MusicStreamingService/blob/b91003c2fdcc51e46b6174bc61f021b53e8f44a7/Dokumentacija-Writerside/dodatni-kod-primjeri/supabase-stripe-schema.sql>)



1 tables created

⚠ Napomena Objašnjenje SQL Koda:

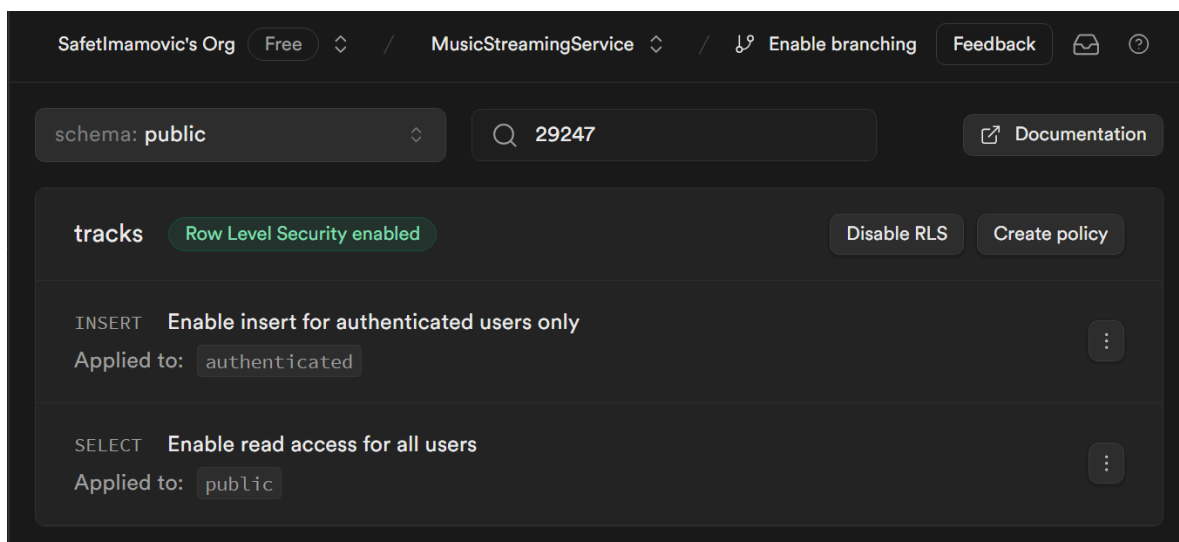
- **Users Tabela:** Čuva podatke o korisnicima sa sigurnošću na nivou reda kako bi se osiguralo da korisnici mogu pregledati i ažurirati samo svoje podatke.
- **Trigger za nove korisnike:** Automatski kreira unos u `users` tabeli kada se novi korisnik registruje.
- **Customers Tabela:** Povezuje ID korisnika sa Stripe ID-om korisnika, nije

dostupno korisnicima.

- **Products Tabela:** Čuva detalje o proizvodima, dostupno svima u read-only načinu.
- **Prices Tabela:** Čuva detalje o cijenama, sinhronizovano sa Stripe-om, dostupno svima u read-only načinu.
- **Subscriptions Tabela:** Čuva detalje o pretplatama, osiguravajući da korisnici mogu pregledati samo svoje pretplate.
- **Realtime Pretplate:** Omogućava real-time praćenje na javnim tabelama `products` i `prices`.

Dodatne Tabele i Politike

1. Tracks Tabela



2 table policies

```
create table public.tracks (  
  id bigint generated by default as identity,  
  created_at timestamp with time zone not null default now(),  
  title text null,  
  track_path text null,  
  image_path text null,
```

```
gif_path text null,  
author  
text null,  
user_id uuid null,  
constraint tracks_pkey primary key (id),  
constraint tracks_user_id_fkey foreign key (user_id) references users  
(id) on delete cascade  
) tablespace pg_default;
```

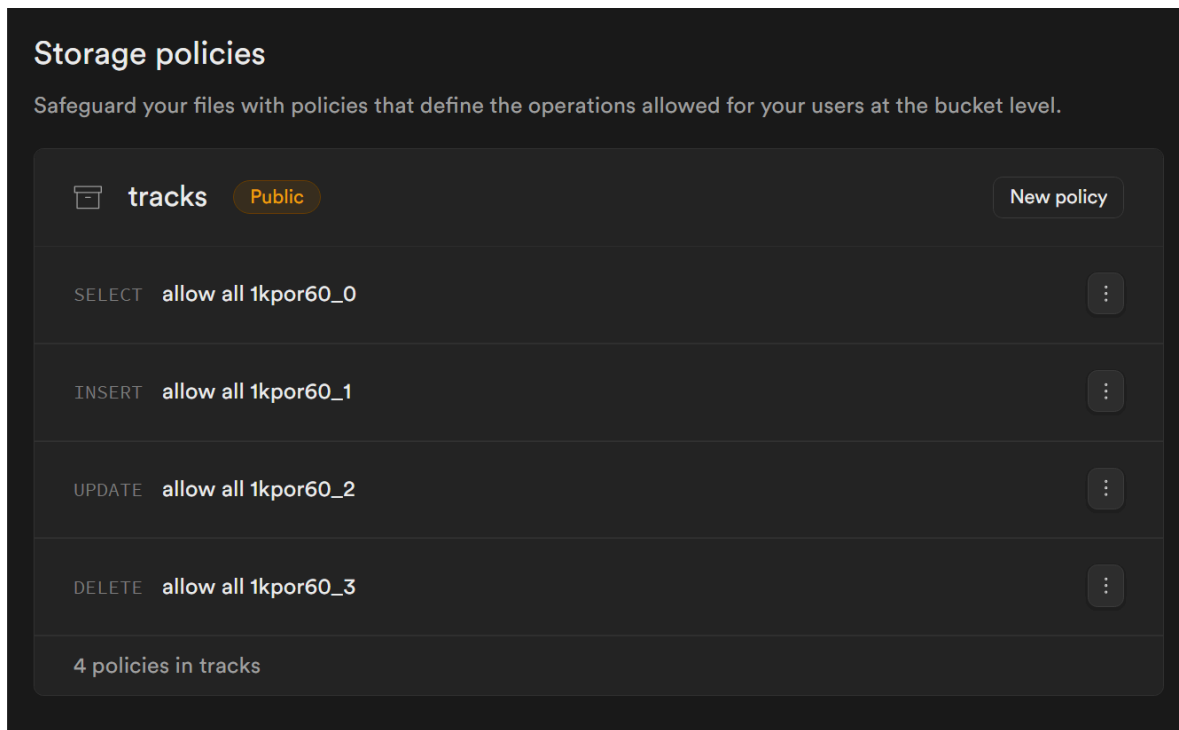
2. Politike za Tracks Tabelu

```
alter policy "Omogući čitanje svim korisnicima" on "public"."tracks" to  
public using (true);  
create policy "Omogući unos samo autentificiranim korisnicima" on  
"public"."tracks" as PERMISSIVE for INSERT to authenticated with check  
(true);
```

3. Tabela Liked Tracks

```
create table public.liked_tracks (  
    user_id uuid not null,  
    created_at timestamp with time zone not null default now(),  
    track_id bigint not null,  
    constraint liked_tracks_pkey primary key (user_id, track_id),  
    constraint liked_tracks_track_id_fkey foreign key (track_id)  
references tracks (id) on delete cascade,  
    constraint liked_tracks_user_id_fkey foreign key (user_id) references  
users (id) on delete cascade  
) tablespace pg_default;
```

4. Politike Tabele Omiljenih Pjesama



3 bucket policies

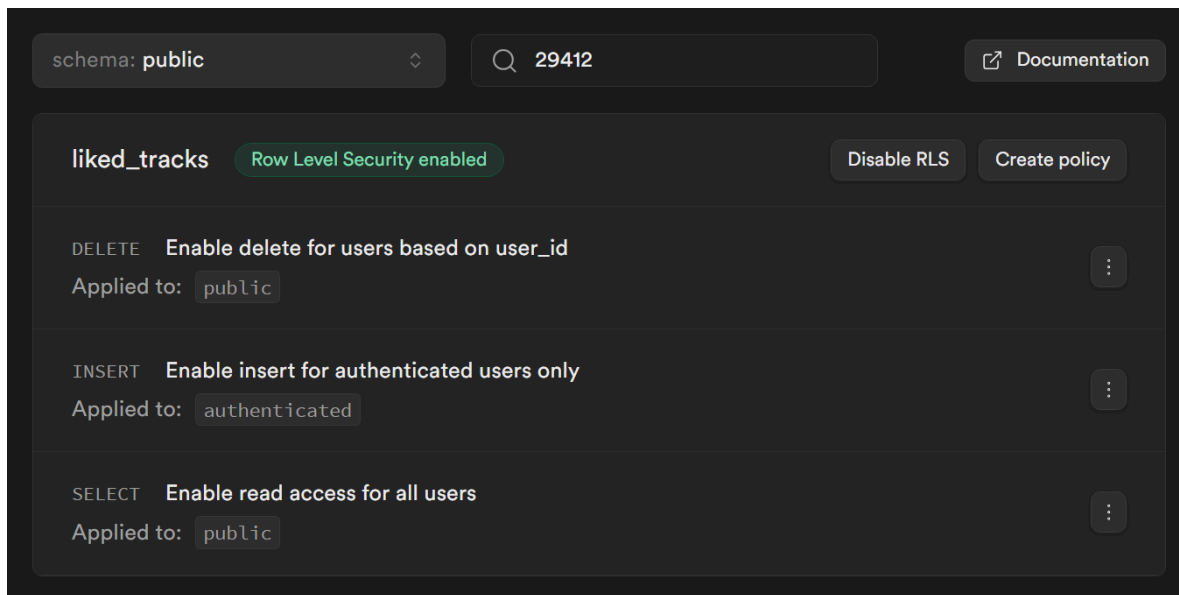
```
alter policy "Dozvoli brisanje korisnicima na osnovu user_id" na
"public"."liked_tracks" za javno korišćenje (((SELECT auth.uid() AS uid)
= user_id));
```

5. Postavljanje Skladišta

6. Kreiranje Kanti

- **Kanta za Pjesme:** Ograničena na audio/mpeg.
- **Kante za Slike i GIF-ove:** Nema ograničenja tipova.

7. Politike Skladišta za Kantu za Pjesme



Liked tracks rls policies

```
CREATE POLICY "Dozvoli sve 1kpor60 0" ON storage.objects FOR SELECT TO public KORIŠTENJEM (bucket_id = 'pjesme');
CREATE POLICY "Dozvoli sve 1kpor60 1" ON storage.objects FOR INSERT TO public SA PROVJEROM (bucket_id = 'pjesme');
CREATE POLICY "Dozvoli sve 1kpor60 2" ON storage.objects FOR UPDATE TO public KORIŠTENJEM (bucket_id = 'pjesme');
CREATE POLICY "Dozvoli sve 1kpor60 3" ON storage.objects FOR DELETE TO public KORIŠTENJEM (bucket_id = 'pjesme');
```

Integracija Supabase-a s TypeScript Tipovima

Korak 1: Instalacija Supabase CLI-a

Za početak, instaliran je Supabase CLI alat kao razvojna zavisnost. Ovaj alat pomaže u generisanju TypeScript tipova iz PostgreSQL šeme.

```
npm i supabase@">=1.8.1" --save-dev
```

Objašnjenje

- `npm i supabase@">=1.8.1" --save-dev`: Instalira se Supabase CLI paket s verzijom većom ili jednakom 1.8.1 kao razvojna zavisnost. `--save-dev` flag osigurava da se koristi samo u razvoju.

Korak 2: Prijava u Supabase

Potrebno je prijaviti se u Supabase kako bi se generisali tipovi. Ovo zahtijeva token za pristup, koji se može generisati sa stranice postavki Supabase projekta.

```
npx supabase login
```

Objašnjenje

- `npx supabase login`: Poziva se unos Supabase tokena za pristup, autentifikujući sesiju i omogućavajući pristup projektu.

Korak 3: Generisanje TypeScript tipova

Sa prijavljenim CLI-jem, generisani su TypeScript tipovi na osnovu PostgreSQL šeme.

```
npx supabase gen types typescript --project-id "$PROJECT_REF" --schema
```



```
public > types_db.ts
```

Objašnjenje

- `npx supabase gen types typescript`: Pokreće se Supabase CLI za generisanje TypeScript tipova.
- `--project-id "$PROJECT_REF"`: Specifikuje ID Supabase projekta.
- `--schema public`: Ukazuje na šemu za koju se generišu tipovi, u ovom slučaju `public` šema.
- `> types_db.ts`: Preusmjerava generisane tipove u fajl pod imenom `types_db.ts`.

Nakon ovog koraka, `types_db.ts` fajl će sadržavati TypeScript tipove za tabele i kolone iz PostgreSQL šeme.

Link do `types_db.ts`

(https://github.com/SafetImamovic/MusicStreamingService/blob/168a6be7bd7ed5989d6ef8bbbee168ac2756eb42/types_db.ts)

Korak 4: Postavljanje Supabase i User Context Providera

4.1: Kreiranje direktorija i fajla

Kreiran je direktorij pod nazivom `providers` i fajl pod nazivom `SupabaseProvider.tsx` u Next.js projektu.

```
mkdir providers
touch providers/SupabaseProvider.tsx
```

4.2: Instalacija Supabase Auth Helpers

Instalirani su potrebni paketi za upravljanje Supabase autentifikacijom u Next.js i React.

```
npm install @supabase/auth-helpers-nextjs @supabase/auth-helpers-react
```

4.3: Implementacija SupabaseProvider

Sljedeći kod je dodan u `SupabaseProvider.tsx` kako bi se kreirao Supabase klijent i obezbijedio kontekst sesije.

(<https://github.com/SafetImamovic/MusicStreamingService/blob/4ac3ec20ea1d576d49a7e1b7a674a972ba0ab0f8/providers/SupabaseProvider.tsx>)

4.4: Omotavanje glavnog layouta sa SupabaseProvider

`layout.tsx` je izmijenjen kako bi uključio `SupabaseProvider`.

(<https://github.com/SafetImamovic/MusicStreamingService/blob/4ac3ec20ea1d576d49a7e1b7a674a972ba0ab0f8/app/layout.tsx>)

Korak 5: Postavljanje User Context-a

5.1: Kreiranje direktorija i fajlova za hookove

Kreiran je direktorij `hooks` i fajl `useUser.tsx`.

```
mkdir hooks
touch hooks/useUser.tsx
```

5.2: Definisanje tipova korisnika

Novi fajl `types.ts` je dodan u korijenu projekta kako bi se definisali TypeScript tipovi za detalje korisnika, proizvode, cijene i pretplate.

(<https://github.com/SafetImamovic/MusicStreamingService/blob/4ac3ec20ea1d576d49a7e1b7a674a972ba0ab0f8/types.ts>)

5.3: Implementacija useUser hook-a

Sljedeći kod je dodan u `useUser.tsx` kako bi se upravljalo korisničkim kontekstom i detaljima pretplate.

(<https://github.com/SafetImamovic/MusicStreamingService/blob/4ac3ec20ea1d576d49a7e1b7a674a972ba0ab0f8/hooks/useUser.tsx>)

5.4: Kreiranje UserProvider-a

Kreiran je fajl `UserProvider.tsx` u `providers` direktoriju i dodan sljedeći kod.

(<https://github.com/SafetImamovic/MusicStreamingService/blob/4ac3ec20ea1d576d49a7e1b7a674a972ba0ab0f8/providers/UserProvider.tsx>)

5.5: Omotavanje layouta sa UserProvider-om

layout.tsx je izmijenjen kako bi uključio UserProvider.

(<https://github.com/SafetImamovic/MusicStreamingService/blob/4ac3ec20ea1d576d49a7e1b7a674a972ba0ab0f8/app/layout.tsx>)

Postavljanje Supabase Zavistnosti

Instalacija Supabase zavisnosti

Instalirano je potrebno pakete za integraciju Supabase autentifikacije u projekt.

```
npm install @supabase/auth-helpers-nextjs  
npm install @supabase/auth-helpers-react
```

2. Supabase Provider

Za konfiguraciju Supabase providera, kreiran je `SupabaseProvider.tsx` u direktoriju `providers`. Ovaj fajl omogućava da cijela aplikacija koristi Supabase instancu.

`SupabaseProvider.tsx`

(<https://github.com/SafetImamovic/MusicStreamingService/blob/17ac6d8a335b225098d42ab5697db2471b5b2982/providers/SupabaseProvider.tsx>)

3. Omotavanje Root Layouta

U root layout-u dodan je `SupabaseProvider` kako bi bila dostupna Supabase instanca u cijeloj aplikaciji.

`layout.tsx`

(<https://github.com/SafetImamovic/MusicStreamingService/blob/17ac6d8a335b225098d42ab5697db2471b5b2982/app/layout.tsx>)

Kreiranje Modalne Komponente

1. Instalacija Radix UI

Za izradu modalnih prozora, instalirani su potrebni paketi Radix UI.

```
npm install @radix-ui/react-dialog
```

2. Modalna Komponenta

Kreirana je modalna komponenta koja koristi Radix UI.

`components/Modal.tsx`

(<https://github.com/SafetImamovic/MusicStreamingService/blob/17ac6d8a335b225098d42ab5697db2471b5b2982/app/components/Modal.tsx>)

[8d42ab5697db2471b5b2982/components/Modal.tsx](https://github.com/SafetImamovic/MusicStreamingService/blob/17ac6d8a335b225098d42ab5697db2471b5b2982/components/Modal.tsx))

Modal za Autentifikaciju

1. Postavljanje Zustanda za upravljanje stanjem modala

Za upravljanje stanjem modala korišten je Zustand.

```
npm install zustand
```

2. Zustand Store za Modal

Kreiran je Zustand store koji upravlja stanjem modalnog prozora.

hooks/useAuthModal.ts

(<https://github.com/SafetImamovic/MusicStreamingService/blob/17ac6d8a335b225098d42ab5697db2471b5b2982/hooks/useAuthModal.ts>)

3. Auth Modal Komponenta

Kreirana je komponenta modalnog prozora za autentifikaciju koja koristi Zustand store.

components/AuthModal.tsx

(<https://github.com/SafetImamovic/MusicStreamingService/blob/17ac6d8a335b225098d42ab5697db2471b5b2982/components/AuthModal.tsx>)

4. Instalacija Supabase Auth UI zavisnosti

Potrebne su zavisnosti za integraciju Supabase Auth UI komponenti.

```
npm install @supabase/auth-ui-react  
npm install @supabase/auth-ui-shared
```

Izmjene na Header-u

1. Izmjena Header-a za pokretanje dijaloga za registraciju

U Header komponenti je napravljena izmjena za pokretanje modalnog prozora za registraciju.

components/Header.tsx

(<https://github.com/SafetImamovic/MusicStreamingService/blob/17ac6d8a335b225098d42ab5697db2471b5b2982/components/Header.tsx>)

[8d42ab5697db2471b5b2982/components/Header.tsx](https://github.com/SafetImamovic/MusicStreamingService/blob/17ac6d8a335b225098d42ab5697db2471b5b2982/components/Header.tsx))

2. Konfiguracija GitHub OAuth-a

Omogućen je GitHub kao provider u Supabase postavkama.

3. Izmjena Header-a za stanje autentifikacije

Izmijenjen je Header kako bi prikazao stanje autentifikacije korisnika.

components/Header.tsx

(<https://github.com/SafetImamovic/MusicStreamingService/blob/17ac6d8a335b225098d42ab5697db2471b5b2982/components/Header.tsx>)

Notifikacije sa React Hot Toast

1. Instalacija React Hot Toast

Za prikaz notifikacija korišten je React Hot Toast.

```
npm install react-hot-toast
```

2. Toaster Provider

Kreiran je `ToasterProvider.tsx` koji omogućava prikazivanje notifikacija u cijeloj aplikaciji.

providers/ToasterProvider.tsx

(<https://github.com/SafetImamovic/MusicStreamingService/blob/17ac6d8a335b225098d42ab5697db2471b5b2982/providers/ToasterProvider.tsx>)

3. Dodavanje ToasterProvider-a u Layout

U Layout komponenti je dodan `ToasterProvider` kako bi notifikacije bile dostupne u cijeloj aplikaciji.

layout.tsx

(<https://github.com/SafetImamovic/MusicStreamingService/blob/17ac6d8a335b225098d42ab5697db2471b5b2982/app/layout.tsx>)

Dodatni Koraci

- Dodan je `<ModalProvider />` u Layout ispod `<UserProvider>` za bolje upravljanje modalnim prozorima.

- Korištena je dokumentacija Radix UI za dodavanje modalnih prozora.

Eksterna HTML Stranica za Kontakt - Slanje na Bazu

Pregled

Ovaj projekt uključivao je kreiranje kontakt forme koja šalje JSON podatke na Supabase server. Projekt je koristio React, Supabase, TypeScript i Tailwind CSS.

Integracija sa Supabase

Vanjska HTML Kontakt Stranica

U našu vanjsku HTML kontakt stranicu uključili smo CDN uvoz za Supabase jer nije bila unutar direktorija komponenti i zahtijevala je slanje JSON podataka na Supabase server.

Dodavanje u ContactUs.html

```
<script src="https://cdn.jsdelivr.net/npm/@supabase/supabase-js@2">
</script>
<script type="module" src="push-kontakt.js"></script>
```

Postavljanje Baze Podataka

Kako bismo pratili kontaktne zahtjeve i poruke primljene od strane servera, kreirana je nova SQL tabela pod nazivom `kontakt_upiti`:

```
create table public.kontakt_upiti (
  id bigint generated by default as identity,
  created_at timestamp with time zone not null default now(),
  ime character varying null,
  prezime character varying null,
  broj_tel character varying null,
  email character varying null,
  poruka text null,
  ip_adresa text null,
```



```
constraint kontakt_upiti_pkey primary key (id)
) tablespace pg_default;
```

JavaScript Logika za Podnošenje Kontakt Forme

Kreiran je novi JavaScript fajl za rukovanje logikom podnošenja kontakt forme i upravljanje varijablama okruženja.

env_vars.js

```
export const url_path = SUPABASE_URL;
export const api_key = SUPABASE_KEY;
```

push-kontakt.js

```
import { createClient } from
'https://cdn.jsdelivr.net/npm/@supabase/supabase-js@2/+esm';
import { url_path, api_key } from './env-vars.js';

document.addEventListener('DOMContentLoaded', function() {

  // Inicijalizacija Supabase klijenta
  const supabase = createClient(url_path, api_key);

  // Event listener za podnošenje forme
  document.getElementById("loginForm").addEventListener("submit",
  async (e) => {
    e.preventDefault();

    // Preuzimanje podataka iz forme
    let ime = document.getElementById("ime").value;
    let prezime = document.getElementById("prezime").value;
    let email = document.getElementById("email").value;
    let telefon = document.getElementById("telefon").value;
    let poruka = document.getElementById("poruka").value;

    // Objekt sa podacima iz forme
```

```

let submit_data = {
  ime: ime,
  prezime: prezime,
  email: email,
  telefon: telefon,
  poruka: poruka
};

for (const key in submit_data) {
  console.log(`${key}: ${submit_data[key]}`);
}

const { error } = await supabase
  .from('kontakt_upiti')
  .insert({
    ime: submit_data.ime,
    prezime: submit_data.prezime,
    email: submit_data.email,
    broj_tel: submit_data.telefon,
    poruka: submit_data.poruka,
    ip_adresa: '1234'
  });

if (error) {
  console.error('Greška prilikom unosa podataka:',
error.message);
} else {
  console.log('Podaci uspješno uneseni:', data);
}
});
});

```

Ova dokumentacija pruža strukturiran pregled koraka poduzetih za integraciju Supabase u projekt, postavljanje baze podataka i implementaciju logike podnošenja kontakt forme.

Inicijalno Postavljanje Layouta

Postavljanje Projekta i Struktura Fajlova

Inicijalno Postavljanje

1. Kreiran `page.tsx`:

- Kreiran u root direktoriju aplikacije.
- Konfigurisan da se prikazuje pri pokretanju stranice koristeći `npm run dev`.

2. Kreiran `site` Folder:

- Smješten u root direktoriju aplikacije.
- Sadrži `page.tsx` fajl.

3. Kreiran `components` Folder:

- Smješten u root direktoriju.
- Pohranjuje sve komponente potrebne za stranicu.

Komponenta Sidebar

1. Kreiran `Sidebar.tsx`:

- Fajl je dizajniran da prikazuje child komponente koristeći sljedeći interfejs i zadatak:

```
interface SidebarProps {
  children: React.ReactNode;
}

const Sidebar: React.FC<SidebarProps> = ({ children }) => {
  return (
    <div>
      {children}
    </div>
  );
}
```

```
    </div>
  );
}

export default Sidebar;
```

2. Kreiran `layout.tsx`:

- Smješten u folderu aplikacije.
- Uvozi `Sidebar.tsx` komponentu koristeći:

```
import Sidebar from '@components/Sidebar';
```

Integracija Ikona

1. Instaliran `react-icons`:

- Korišten za integraciju ikona u sidebar i druge komponente.
- Instaliran pomoću sljedeće komande:

```
npm install react-icons
```

2. Uvezene Ikone:

```
import { HiHome } from "react-icons/hi";
import { BiSearch } from "react-icons/bi";
```

Komponenta Box

1. Kreiran `Box.tsx`:

- Smješten u `components` folderu.
- Koristi `tailwind-merge` za proširenje mogućnosti Tailwind CSS-a.

2. Instaliran `tailwind-merge`:

- Instaliran pomoću sljedeće komande:

```
npm install tailwind-merge
```

3. Kod Komponente `Box`:

```
import { twMerge } from "tailwind-merge";

interface BoxProps {
  children: React.ReactNode;
  className?: string;
}

const Box: React.FC<BoxProps> = ({ children, className }) => {
  return (
    <div
      className={twMerge(
        `bg-neutral-900 rounded-lg h-fit w-full`,
        className
      )}
    >
      {children}
    </div>
  );
}

export default Box;
```

Dodatne Komponente

1. Kreiran `SidebarItem.tsx`:

- Smješten u `components` folderu.
- Povezan sa `Sidebar.tsx` koristeći:

```
import SidebarItem from "../SidebarItem";
```

2. Kreiran **Library.tsx**:

- Smješten u **components** folderu.
- Koristi se za upload i pohranu pjesama, te upravljanje playlistama.

3. Kreiran **Header.tsx**:

- Smješten u **components** folderu.
- Koristi se u **page.tsx**.

4. Kreiran **Button.tsx**:

- Smješten u **components** folderu.
- Koristi se za kreiranje dugmadi za aplikaciju.

Slike

- Dodana slika za dugme za omiljene pjesme:

```
https://i1.sndcdn.com/artworks-y6qitUuZoS6y8LQo-5s2pPA-t500x500.jpg
```

Završna Implementacija page.tsx

Na kraju razvoja, **page.tsx** fajl je izgledao ovako:

```
import Header from "@components/Header";
import ListItem from "@components/ListItem";


export default function Home() {
  return (
    <div className="
      bg-neutral-900
      rounded-lg
      h-full
```




```
        </div>
      </div>
    </div>
  )
}
```


Napomene

Nema eksplicitnih napomena za ovaj projekat. Sve napomene sto se tiču koda su oznacene:

 Napomena: Ovo je napomena

 Ovo je napomena