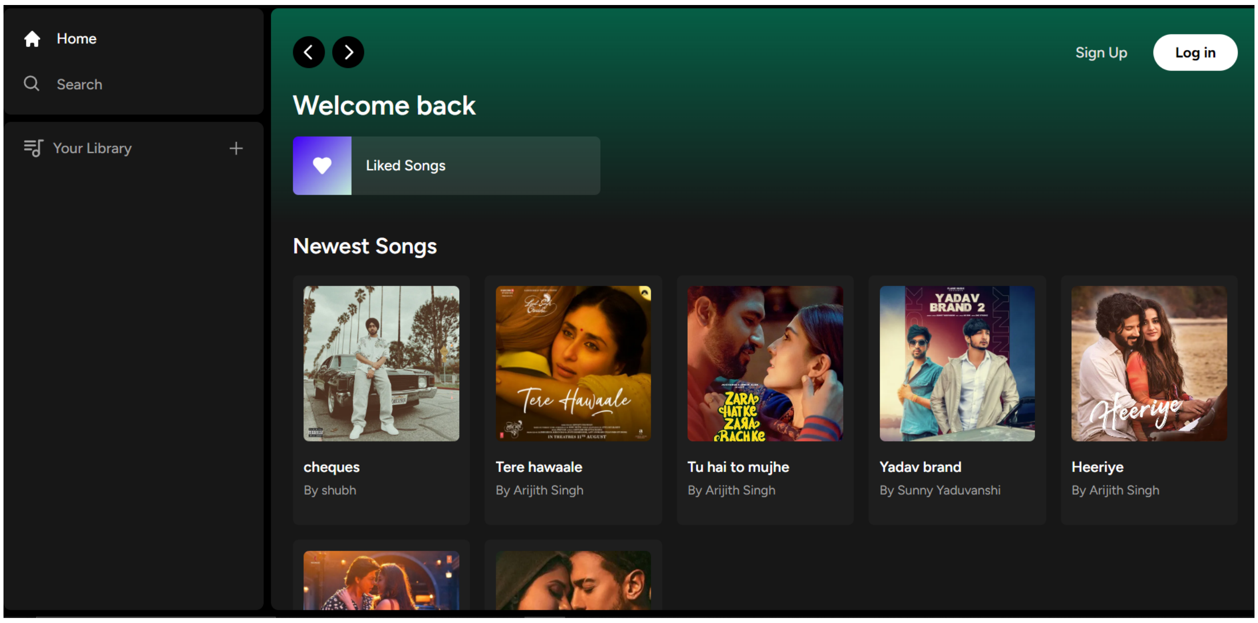

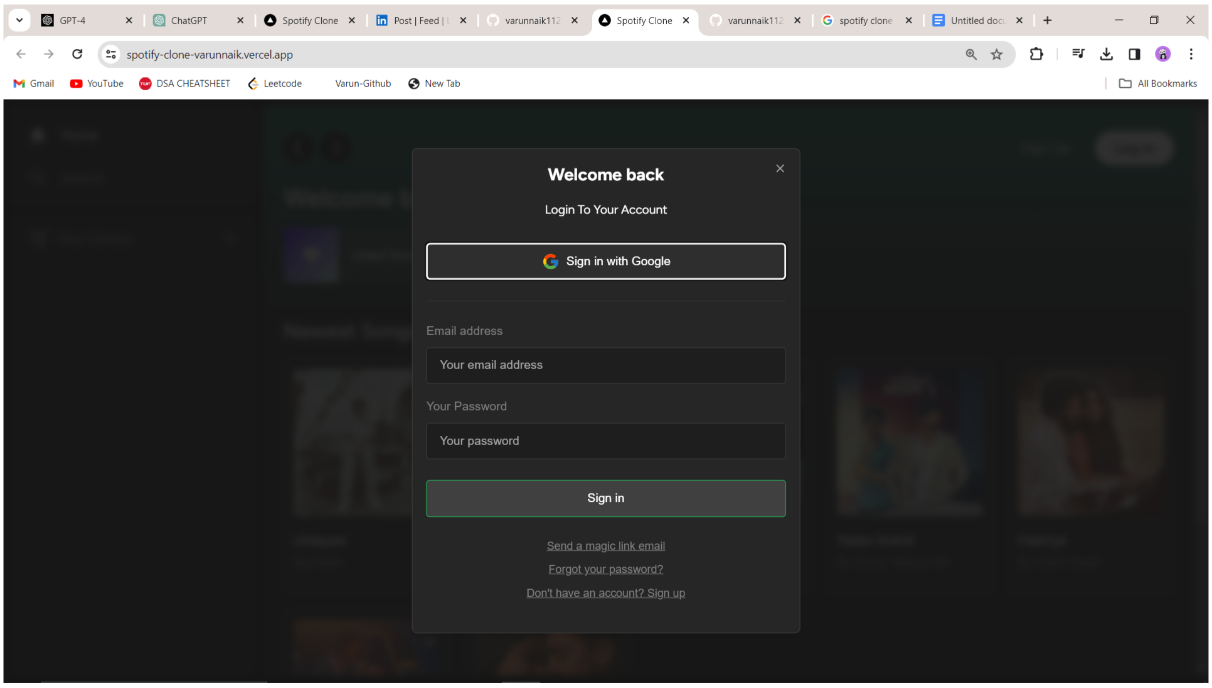

MAJOR PROJECT DETAILS (WEB DEVELOPMENT)

SPOTIFY-CLONE

NAME:VARUN P
Major project:

Technologies used nextjs typescript tailwind css supabase ..
Github link:https://github.com/varunnaik1121/spotify-clone-typescript

Added only some important layout codes because the project is done using nextjs so difficult to put all the code here.

```
import { Figtree } from 'next/font/google';
import './globals.css';
import Sidebar from '@/components/Sidebar';
import SupabaseProvider from '@/providers/SupabaseProvider';
import UserProvider from '@/providers/UserProvider';
import ModalProvider from '@/providers/ModalProvider';
import ToasterProvider from '@/providers/ToasterProvider';
import getSongsByUserId from '@/actions/getSongsByUserId';
import Player from '@/components/Player';
import getActiveProductsWithPrices from
'@/actions/getActiveProductsWithPrices';
const font = Figtree({ subsets: ['latin'] });

export const metadata = {
  title: 'Spotify Clone',
  description: 'Listen to music',
};
export const revalidate = 0;

export default async function RootLayout({
  children,
}: {
  children: React.ReactNode;
}) {
  const userSongs = await getSongsByUserId();
  const products = await getActiveProductsWithPrices();

  return (
    <html lang="en">
      <body className={font.className}>
        <ToasterProvider />
        <SupabaseProvider>
          <UserProvider>
            <ModalProvider products={products} />
            <Sidebar songs={userSongs}>{children}</Sidebar>
            <Player />
          </UserProvider>
        </SupabaseProvider>
      </body>
    </html>
  );
}

import getSongs from '@/actions/getSongs';
import Header from '@/components/Header';
import ListItem from '@/components/ListItem';
import PageContent from './components/PageContent';
export const revalidate = 0;
export default async function Home() {
  const songs = await getSongs();

  return (
    <div className="bg-neutral-900 rounded-lg h-full w-full
overflow-hidden overflow-y-auto">
      <Header>
        <div className="mb-2">
          <h1 className="text-white text-3xl font-semibold">Welcome
back</h1>
          <div className="grid grid-cols-1 sm:grid-cols-2
xl:grid-cols-3 2xl:grid-cols-4 gap-3 mt-4">
            <ListItem
              image={'/images/liked.png'}
              name={'Liked Songs'}
              href={'liked'}
            />
          </div>
        </div>
      </Header>
      <div className="mt-2 mb-7 px-6">
        <div className="flex justify-between items-center">
          <h1 className="text-white text-2xl font-semibold">Newest
Songs</h1>
        </div>
        <PageContent songs={songs} />
      </div>
    </div>
  );
}
```

```
'use client';
import {
  useSupabaseClient,
  useSessionContext,
} from '@supabase/auth-helpers-react';
import { useRouter } from 'next/navigation';
import Modal from './Modal';
import { Auth } from '@supabase/auth-ui-react';
import { ThemeSupa } from '@supabase/auth-ui-shared';
import useAuthModal from '@/hooks/useAuthModal';
import { useEffect } from 'react';

const AuthModal = () => {
  const supabaseClient = useSupabaseClient();
  const router = useRouter();
  const { session } = useSessionContext();

  const { onClose, isOpen } = useAuthModal();
  useEffect(() => {
    if (session) {
      router.refresh();
      onClose();
    }
  }, [session, router, onClose]);

  const onChange = (open: boolean) => {
    if (!open) {
      onClose();
    }
  };
  return (
    <Modal
      title="Welcome back"
      description="Login to your account"
      isOpen={isOpen}
      onChange={onChange}
    >
      <Auth
        theme="dark"
        magicLink
        providers={['google']}
        supabaseClient={supabaseClient}
        appearance={{
          theme: ThemeSupa,
          variables: {
            default: {
              colors: {
                brand: '#404040',
                brandAccent: '#22c55e',
              },
            },
          },
        }}
      />
    </Modal>
  );
};

export default AuthModal;
```

```
'use client';

import { HiHome } from 'react-icons/hi';
import { BiSearch } from 'react-icons/bi';
import { twMerge } from 'tailwind-merge';
import { usePathname } from 'next/navigation';

import SidebarItem from './SidebarItem';
import Box from './Box';
import Library from './Library';
import { useMemo } from 'react';
import { Song } from '@/types';
import usePlayer from '@/hooks/usePlayer';

interface SidebarProps {
  children: React.ReactNode;
  songs: Song[];
}

const Sidebar = ({ children, songs }: SidebarProps) => {
  const pathname = usePathname();
  const player = usePlayer();

  const routes = useMemo(
    () => [
      {
        icon: HiHome,
        label: 'Home',
        active: pathname !== '/search',
        href: '/',
      },
      {
        icon: BiSearch,
        label: 'Search',
        href: '/search',
        active: pathname === '/search',
      },
    ],
    [pathname]
  );

  return (
    <div
      className={twMerge(
        `flex h-full `,
        player.activeId && 'h-[calc(100%-80px)]'
      )}
    >
      <div
        className="
          hidden
          md:flex
          flex-col
          gap-y-2
          bg-black
          h-full
          w-[300px]
          p-2
        "
      >
        <Box>
          <div className="flex flex-col gap-y-4 px-5 py-4">
            {routes.map((item) => (
              <SidebarItem key={item.label} {...item} />
            ))}
          </div>
        </Box>
        <Box className="overflow-y-auto h-full">
          <Library songs={songs} />
        </Box>
      </div>
      <main className="h-full flex-1 overflow-y-auto
py-2">{children}</main>
    </div>
  );
};

export default Sidebar;
```

```
'use client';
import useUploadModal from '@/hooks/useUploadModal';
import Modal from './Modal';
import { FieldValues, useForm, SubmitHandler } from 'react-hook-form';
import { useState } from 'react';
import Input from './Input';
import uniqid from 'uniqid';
import Button from './Button';
import toast from 'react-hot-toast';
import { useUser } from '@/hooks/useUser';
import { useSupabaseClient } from '@supabase/auth-helpers-react';
import { useRouter } from 'next/navigation';
const UploadModal = () => {
  const router = useRouter();
  const [isLoading, setIsLoading] = useState(false);
  const uploadModal = useUploadModal();
  const { user } = useUser();
  const supabaseClient = useSupabaseClient();
  const { register, handleSubmit, reset } = useForm<FieldValues>({
    defaultValues: {
      author: '',
      title: '',
      song: null,
      image: null,
    },
  });
  const onChange = (open: boolean) => {
    if (!open) {
      reset();
      uploadModal.onClose();
    }
  };
  const onSubmit: SubmitHandler<FieldValues> = async (values) => {
    try {
      setIsLoading(true);
      const imageFile = values.image?.[0];
      const songFile = values.song?.[0];
      if (!imageFile || !songFile || !user) {
        toast.error('Missing fields');
        return;
      }
      const uniqueID = uniqid();
      //Uplaod song
      const { data: songData, error: songError } = await
supabaseClient.storage
        .from('songs')
        .upload(`song-${values.title}-${uniqueID}`, songFile, {
          cacheControl: '3600',
          upsert: false,
        });

      if (songError) {
        setIsLoading(false);
        return toast.error('Failed song upload.');
      }
      //Upload image

      const { data: imageData, error: imageError } =
        await supabaseClient.storage
          .from('images')
          .upload(`image-${values.title}-${uniqueID}`, imageFile, {
            cacheControl: '3600',
            upsert: false,
          });

      if (imageError) {
        setIsLoading(false);
        return toast.error('Failed image upload.');
      }
      const { error: supabaseError } = await supabaseClient
        .from('songs')
        .insert({
          user_id: user.id,
          title: values.title,
          author: values.author,
          image_path: imageData.path,
          song_path: songData.path,
        });

      if (supabaseError) {
        setIsLoading(false);
        return toast.error(supabaseError.message);
      }

      router.refresh();
      setIsLoading(false);
      toast.success('Song created!');
      reset();
      uploadModal.onClose();
    } catch (error) {
      toast.error('Something went wrong');
    } finally {
      setIsLoading(false);
    }
    //upload to supabase
  };

  return (
    <Modal
      title="Add a song"
      description="Upload an Mp3 File"
      isOpen={uploadModal.isOpen}
      onChange={onChange}
    >
      <form onSubmit={handleSubmit(onSubmit)} className="flex flex-col
gap-y-4">
        <Input
          id="title"
          disabled={isLoading}
          {...register('title', { required: true })}
          placeholder="Song title"
        />
        <Input
          id="author"
          disabled={isLoading}
          {...register('author', { required: true })}
          placeholder="Song author"
        />
        <div>
          <div className="pb-1">Select a song file</div>
          <Input
            id="song"
            type="file"
            disabled={isLoading}
            {...register('song', { required: true })}
            accept=".mp3"
          />
        </div>
        <div>
          <div className="pb-1">Select an image</div>
          <Input
            id="image"
            type="file"
            disabled={isLoading}
            {...register('image', { required: true })}
            accept="image/*"
          />
        </div>
        <Button disabled={isLoading} type="submit">
          Create
        </Button>
      </form>
    </Modal>
  );
};

export default UploadModal;
```

THANK YOU