1 Inledning

1.1 Kort beskrivning av projekt

Jag vill göra en webbsida som man kan streama musik / podcasts på via youtube. Användare ska kunna skapa ett konto på sidan, scrolla igenom topplistor och lägga låtar i spellistor samt söka efter artister och album. Man ska såklart också kunna spela upp låtar och kanske även se låttexten live. Det skulle vara coolt att ha ett kommentar-system likt soundcloud, men det är en extrasak som jag kanske kan göra om jag har tid över. Jag ska också ha med en specifik podcast-del av sidan där man kan bläddra mellan alla podcasts som är uppladdade på youtube

1.2 Beskrivning av målgrupp

Min målgrupp består av användare som letar efter ett alternativ till youtube music och spotify. Jag vill se till att ha med fler podcasts än spotify så att min app kan bli introducerad till folk som letar efter en ny podcast-app.

1.3 Vilket betyg satsar du på

Med det här projektet satsar jag på ett A.

2 Ingående beskrivning av projektet

2.1 Beskrivningen

Projektets frontend kommer att skrivas i html css och javascript, och kommer att interagera med backend för att calla requests till olika api:er såsom youtubes egna som en sökmotor och ytdl för streaming av video. På sidan ska användare kunna registrera och logga in, spela låtar samt skapa spellistor som andra kan ha tillgång till.

2.2 Lista allt som skall göras för att projektet skall bli till

Detta bör fördelas över rubrikerna nedan:

2.2.1 – Frontend

- login register frontend
- topplista
    - via youtubes egna api
- search
    - yt api
- artist profile
- album / playlist list

- player styling
- mycket css

2.2.2 – Backend

- säker login / register med krypterade lösenord och sessions
    - använder paketet bcrypt och express + express-sessions
- interaktion med youtubes api för att söka
    - client →(‘sökning’) backend →(req med massa info) youtube
    - för att kunna få info från youtube behövs ett “service konto” och ett projekt skapat med google api
- streaming av ljud från youtube via ytdl
    - får bara användas av användare med valid session id
    - streaming, session, routing

2.2.3 – Databas

databas med mysql

Users

- user_id (Primär Nyckel), username, password, email, created_at

Playlists

- playlist_id (Primär Nyckel), user_id (Utomstående Nyckel), name, description, is_public, created_at
- Utomstående Nyckel: user_id (Refererar till Users.user_id)

Playlist

- playlist_id (Utomstående Nyckel), song_data, added_at
- Utomstående Nyckel: playlist_id (Refererar till Playlists.playlist_id)

2.2.4 – Om du har något som inte passar in i de ovan.

- https://www.npmjs.com/package/ytdl-core (streaming)
- https://www.npmjs.com/package/fluent-ffmpeg (vid till mp3)
- https://ytmusicapi.readthedocs.io/en/stable/usage.html (api till youtube)

2.3 Beskriv vilka sidor/html-dokument/script-dokuemnt etc som webbplatsen kommer att använda sig av samt andra speciella saker du tycker är värt att nämna.

Denna fråga är i princip de saker du listade under 2.2 men faktiskt fördelat på olika dokument samt mer konkreta beskrivningar av dessa dokument. Se gärna exempelplaneringen i Classroom för inspiration hur denna del kan vara uppbyggd.

frontend - html

auth
sida med inlogg och registrering
index
"huvud" sidan med ett modulärt system där användare kan stänga av och flytta element i en lista som sedan flyttas på sidan. som en navbar fast istället för länkar är det knappar som man kan trycka på
search
sida med searchbar och resultat. kopplat med youtubes api och egna databasen för att både visa låtar, artister och user-generated playlists.
playlists
sida för individuella playlists med olika information så som bild, descriiption ocjh titel samt lista för alla låtar. autogenererad av backend.

frontend - js

Jag kommer att ha massa olika funktioner som genererar element och hanterar requests till backend.

index - navbar
jag behöver många eventlisteners för alla delar i min "nav bar" funktionerna kommer vara *handle_drag_start, handle_drag_over, handle_drag_enter, handle_drag_leave, handle_drag_end, handle_drop*
handle drop hanterar vad som ska hända när användare har plockat upp ett element och ser till att elementet sätter sig på rätt plats, samt spelar ändringen till resten av sidan.
navbaren ska inte bara hantera positionen av element, utan ska även hantera ifall elementen ska finnas. för det så har jag handle_main_content, och en till 'onclick' eventlistener på varje element. när man klickar på något så läggs en klass till på nav-item:en för att visa för användaren att ändringen skedde. Navbarens position on status ska sparas på något sätt, och då väljer jag över localstorage eller via databasen.
index - showcase
en av elementen på sidan ska visa upp vilken sång som spelas. här vill vi uppdatera information om sången, och också ta fram en "accent color" från sångens cover art som vi sedan använder i vår css. informationen tar vi från playern, som tas från servern.
search - kommunikation med backend
vi har 2 funktioner som har med backend att göra som skickar ut requests. den ena, search youtube anropar den andra, authenticate and execute, som gör själva post-requesten till vår backend. svaret tas sedan tillbaka och search youtube anropar en till funktion för display.
search - display

här tar vi listan med sökresultat, loopar igenom den och lägger ut element på webbsidan med information. informationen innehåller titel, artist, datum och cover art. vi lägger också till en eventlistener till varje barn i listan som skickar ut ett custom event när användaren klickar på en låt, som har barnets information som beskrivning. det här gör vi för att kunna komma åt informationen i andra filer.

player

jag ska också skriva en egen musikspelare. här tar vi hand om playstate, que och all information om låten.

backend - js

I vår backend så kommer vi ha 3 olika funktioner; en som söker låtar, en som spelar låtar och en som autentiserar användare.

2.4 Moduler

lista på moduler:

- login /register
- search function
- music player
- que “manager”
- skapa playlists
- söka igenom playlists
- explore
- artist profiles

Jag tror att jag kommer hinna göra klart alla basfunktioner såsom att söka och spela låtar. Extrasaker kan vara filter i sökfunktioner, artist profiles och explore que; de bygger på den tidigare koden.

2.5 Databas

Här bör du lägga in modell med relationer för din databas (se datamodellings-dokumentet) samt bilder på tabellerna (se till att se över normalisering i datamodellingsdokumentet för att minska risken att du gör fel).

2.6 Kunskaper

Jag behöver lära mig mycket om hur man skriver en backend och hur man hanterar api:er från en server. sessions, cookies och en stor del av backendkoden är nytt för mig.

2.7 Säkerhet

När du jobbar med backend och databaser till skillnad från bara HTML och CSS så finns det också potentiella säkerhetsrisker. Till exempel skulle lösenord skrivna i klarspråk i databasen utan någon kryptering kunna utgöra en säkerhetsrisk. Vilka potentiella säkerhetsrisker kan du identifiera i ditt projekt och vilka lösningar har du för dessa?
Jag sparar mina lösenord efter kryptering via bcrypt och gör alla requests via min backend så att api keys inte blir exposed. Jag sparar också min session med cookies så att användaren förblir inloggad och ska regenerera kakorna ibland så att det inte kan användas för att få tillgång till sidans funktioner av någon annan.

3 Tidsplanering

3.1 Deadlines

vecka 1
- skiss på websida

vecka 2
- frontend websida

vecka 3
- backend search, streaming

vecka 4
- music player

vecka 5
- spellistor

vecka 6
- databas