Yousra Diab

FURPS RAPPORT – ARITST PP

Projektoversigt:

Projektet er en full-stack web-applikation, der skal give brugere mulighed for at administrere og udforske information om musikkunstnere. Applikationen omfatter både en backend og frontend, udviklet med moderne teknologier og programmeringsprincipper. Koblingen mellem backend og frontend er et REST API.

Functionality (Funktionalitet):

Full-stack web-applikation med følgende krav:

- **Backend**
    - **REST API implementeret med Node.js og Express.js**
        - Routes med endpoint for HTTP-metoderne GET, POST, PUT/PATCH, DELETE.
        - CRUD-operationer der læser og skriver til JSON-fil.
        - Det skal være muligt at kunne få både alle objekter og et objekt på baggrund af specificeret id.
    - **JSON-fil som datakilde**
        - Artist-objekterne i JSON-listen skal bestå af minumum følgende properties: name, birthdate, activeSince, genres, labels, website, image, shortDescription
- **Frontend**
    - User Interface implementeret med HTML, CSS og JavaScript
    - Brugeren skal kunne oprette, læse, opdatere og slette data (CRUD)
    - Filtrering og sortering på udvalgte parametere (props)
    - Du skal kunne markere en kunstner som favorit og vise en liste over alle favoritkunstnere. Du bestemmer selv, hvordan denne liste gemmes (backend, localStorage, variabel eller lignende).
    - Anvendelse af CSS Grid, CSS Flex og/eller HTML Table samt relaterede HTML-elementer
    - Kode opdelt i modules

Usability (Brugervenlighed):

- **Brugergrænseflade:**

Yousra Diab

HTML-siden skal indeholde en overskuelig brugergrænseflade med navigation, formularer og knapper til at interagere med kunstnerne. Brugere kan nemt navigere og udføre opgaver som oprettelse, opdatering og sletning af kunstnere. Brug af dialogelementer til oprettelse, opdatering og sletning af kunstnere giver en interaktiv og brugervenlig oplevelse.

Reliability (Pålidelighed):

Koden skal generere en brugergrænseflade (UI) til at vise kunstneroplysninger, herunder navn, billede, fødselsdato, aktiv siden, genre, label, webside og kort beskrivelse. Brugerne kan interagere med UI-elementer som knapper for opdatering, sletning og favoritter.

- **REST API Pålidelighed:**

REST API'et skal indeholde fejlhåndtering og giver feedback til klienter ved fejl, f.eks. når en kunstner ikke findes

- **Fejlhåndtering:**

Koden har indbygget fejlhåndtering og returnerer fejlstatuskoder og meddelelser, når der opstår fejl, f.eks. når en kunstner ikke findes. Fejlhåndteringsredskaberne skal hjælpe med at sikre, at brugerinput er gyldigt, og at serveranmodninger behandles korrekt. Koden skal indeholde en bekræftelsesdialog, når brugere forsøger at slette en kunstner.

Performance (Ydeevne):

- **Ydeevnekrav:**

Koden indeholder ikke specifikke ydeevnekrav. Ydeevne kan variere afhængigt af serverens responstid og netværksforhold.

Supportability (Support):

- **Vedligeholdelse:**

Koden skal indeholde en metode til at opdatere visningen efter oprettelse, opdatering og sletning af kunstnere. Koden bruger en JSON-fil til at gemme kunstnerdata, hvilket gør det relativt enkelt at vedligeholde og opdatere dataene. Der er potentiale for fremtidig udvidelse af API'et med nye funktioner, hvis kravene ændres.