**Functional:** Applikationen skal opfylde kravene for CRUD-operationer, filtrering, sortering og præsentation af kunstnerdata. Funktionaliteten skal være pålidelig og intuitiv.

**Løsning:** Når applikationen åbnes, og databasen kører, **vil** hjemmesiden automatisk indlæse de forskellige artister **i** databasen. **Herefter er det** muligt for brugeren **at** både ”Create”, ”Update” **og** ”Delete” de forskellige artister. I toppen af skærmen er der et **fieldset**, som indeholder de forskellige muligheder for filtrering og sortering.

**Usability:** Brugergrænsefladen skal være brugervenlig og nem at navigere. Alle CRUD-operationer skal være let tilgængelige, og filtrerings-/sorteringsfunktionerne skal være intuitive.

**Løsning:** Når brugeren åbner hjemmesiden, **vil de blive mødt af en mere farverig side, men samtidig rolig nok til ikke at stresse brugeren. Det** vil være nemt for brugeren **at få et hurtigt overblik** over, hvad de kan gøre på siden. Både filter/sorterings delen, oprettelse **og manipulation** af de forskellige artister **fremgår** tydeligt og håndterligt.

**Reliability:** Applikationen skal være stabil og pålidelig. Dataintegritet og korrekt håndtering af CRUD-operationer er afgørende.

**Løsning:** Ud fra **afprøvning** af applikationen og hjemmesiden **sammen fremgår det**, at håndteringerne **og udførelsen** af handling **sker hurtigt og pålideligt**.

**Performance:** Applikationen skal have en acceptabel ydeevne, herunder hurtig datahentning og responsivitet i Brugergrænsefladen.

**Løsning:** Eftersom applikationen og hjemmesiden **er gemt lokalt**, så er meget **af ydeevnen og hastigheden** af handling **påvirket af brugerens hardware**. Men selv med disse restriktioner **er** applikationen stadigvæk hurtig og responsiv.

**Supportability:** Koden skal være velstruktureret og veldokumenteret, så det er nemt for fremtidige udviklere af vedligeholde og udvide applikationen.

**Løsning:** I koden **er** der blevet lavet forskellige markeringer for at signalere til udviklerne, hvor forskellige kode handlinger skal foregå. Samtidig er forskellige funktioner og handlinger **blevet beskrevet med korte forklarende overskrifter.**

**+:** Der skal også tages hensyn til andre relevante krav og overvejelser for at sikre en vellykket udvikling og implementering af applikationen. Ud over FURPS-kravene skal udviklingen også overholde generelle principper for kodestruktur, modularitet og afhængighedsstyring, herunder Separation of Concerns, Loose Coupling og High Cohesion.

**Løsning:** Meget af koden **er** opdelt i forskellige JavaScript filer, **som** så **er** navngivet efter deres funktionalitets område. Dette gør, at koden **er** mere modulær, **og en udefrakommende udvikler vil derfor** nemt kunne skabe **sig** et overblik over, hvor givende funktionalitet kan være placeret. Dette gør også, at der er en højere form af kohæsion mellem, hvad der står i hvilke filer. Ved at gøre det på denne måde skabes der også en separation of concerns, samt loose coupling**, hvilet vil sige, at når funktioner kalder andre funktioner, så kalder de dem kun i én retning, indtil funktionen har nået deres ende.**