FURPS+ Report for Music Artist Database Application

A brief overview of the project: Requirements and objectives.

The objective of the project is to develop a comprehensive fullstack application that functions as a music artist database. The term "fullstack" in this context implies that both the frontend and backend components will be developed internally. The primary goals of the project are to create a user-friendly, reliable, and performant application that meets the following requirements and priorities:

The application must meet the requirements of CRUD operations, an example of filtering, sorting and presentation of the artist data. The functionality must be reliable and intuitive.

The UI must be user-friendly and easy to navigate. The CRUD operations must be easy to use, and the filter/sort functions must be intuitive. The app in itself must be stable and reliable. The integrity of the data and use of CRUD is vital for the usability of the app.

The app must have an acceptable performance, therein be fast at retrieving data and have a responsive UI. The code must be well structured and documented, so it would be easy for future developers to maintain and expand the app.

Functionality:

Backend

- REST API implemented with the use of Node.js, and the crucial dependency of Express.js
    - Routes at the endpoint for GET, POST, PUT/PATCH and DELETE (CREATE, READ, UPDATE and DELETE)
    - JSON files are used as the data for these CRUD operations
    - The following properties are used for the JSON objects:
        - Name, birthdate, activeSince, genres, labels, website, image, shortDescripton
        - ID is created with the Node dependency, UUID.

Frontend

- UI is created by using HTML, CSS and JS.
- All CRUD operations are accessible
- Filtering and sorting on certain properties

- A favorite list is saved as localStorage, for future usage. This list can be used to filter the shown artists.
- The code is built into two modules: REST and everything else.

Usability

Sorting, Filtering, and CRUD Operations: Sorting, filtering, and all CRUD operations have been designed with a user-centric approach, prioritizing ease of use and navigation.

Reliability

Data Formatting: Prior to transmission between the client and server, all data involved in CRUD operations undergoes rigorous formatting to ensure reliability.

Performance

Fast and Modern Performance: The application aims to deliver a fast and modern performance, which is particularly reflected in the responsiveness of the user interface.

Supportability

Modular Codebase: The codebase is structured into two main modules, facilitating loose coupling and separation of concerns, which contributes to supportability and maintainability.