JavaScript Programmer Task [IIT Bombay] by Nayana R

Design Document

-Overview

The project is a dynamic web application that allows users to manage a list of chemical supplies in a tabular format. Users can add, edit, delete, and sort the entries, providing a straightforward interface for data manipulation.

-Design Goals

Ensure that the application is easy to use, with intuitive controls for editing and managing data.

-Architecture

--Structure

---**HTML:** Defines the layout and structure of the user interface.

---**CSS:** Styles the application, ensuring it is visually appealing and user-friendly.

---**JavaScript:** Implements the functionality, including data manipulation, event handling, and dynamic updates to the DOM

--Data Management

The application manages data in an array of objects stored in the file data.js, where each object represents a chemical supply item. This choice allows for easy data manipulation (adding, editing, and deleting items) and provides a clear mapping between the data model and the table displayed in the user interface.

--Key Features

---**Dynamic Table Generation:** The **generateTable** function is responsible for creating the table and populating it with data. The design choice to dynamically generate the table based on the data array allows for easy updates and ensures

that any changes to the data are reflected immediately in the user interface.

---Row Highlighting

When a user clicks on a row, it gets highlighted, providing visual feedback. The user can navigate up and down the table by using the up and down arrow button.

---Row Deletion

After selecting a row, the user can easily delete that particular row by clicking on the delete button. The row is removed from the table

---Sorting Functionality

The table can be sorted by clicking on the header of each column. This is implemented using a sort order object that toggles between ascending and descending states. This feature enhances user experience by allowing quick access to relevant data.

---Reset Functionality

After sorting the table by clicking on any number of header, if the user wishes to see the table in the original form than the user can click on the reset button to reset the table contents to its original form.

---Row Editing

Double-clicking on a cell enables editing, allowing for in-place updates. This design choice minimizes disruption, as users can edit data directly within the context of the table. After inputting the new value, the user MUST click on the save button. Please note that different cells of the same row can be edited at a single time, but not different rows.

---Adding Data (Adding a new Row)

The user can facilitate adding new data by clicking on the plus button. This adds a new row at the end of the table, after which the user can double click on each cell to add new

value. To ensure the data is added the user MUST click on the save button.

---Input Handling

The application uses input elements for editing. To maintain a clean user experience, input fields are displayed only when editing. The cell returns to its original state on blur, ensuring that the user can easily navigate through the table without unintended disruptions.

-Future Enhancements

--Data Persistence: Implement local storage or database integration to persist data across sessions.

-Conclusion

This project was designed with a focus on usability, maintainability, and responsiveness. The decisions made throughout the development process aim to create a seamless experience for users while ensuring that the codebase remains clean and easy to understand.

[Disclaimer: Some sections of the code and design were written with assistance from ChatGPT, an AI language model developed by OpenAI. The AI was used as a tool for solving technical challenges, and for improving code efficiency. All final decisions regarding the design and implementation were made by the author, ensuring that the code meets the project's specific requirements.]