**Project Title:** 240912\_Task\_JS\_Programmer

**Project Description:**  
This project is given to me as a task to create a table where I can create, update, and delete them, performing all the CRUD operations in HTML, CSS, and Java-script languages. This code also rearranges all the data in ascending or descending order, clicking by column name.

**Adding new row data:**

We can add new row by clicking on the plus sign from the right side of the table. When the page loads, we can add the specific details about the chemical supplies in the text boxes and then submit the form; it will redirect to the chemical supplies tables after saving the data.

**Updating the existing columns:**

If we want to edit density, viscosity, and quantity columns, we can directly add new values in the given text box in the row, and then click on a Save icon from the right side, the data will be reflected in the table.

If we want to edit the other columns rather than Density, Viscosity, and Quantity, or if we want to update all the given columns for the specific row, then we can double-click on it, and it will redirect to the editing form, adding the values we can submit the form, and then it will redirect to the chemical supplies table with the updated data.

**Deleting the existing rows:**

If we want to delete the rows, then we can select the row that we want to delete. After selecting the row, the blue color will appear on the row, then we can click on the dustbin icon, and the row will be deleted from the existing table.

**Re-arranging the existing columns:**

If we want to re-arrange the columns in the ascending or descending format, we can click on the column name, and it will re-arrange the data for the existing table.

**How to install and run the project:**

Please click on the below link for the installment of the project and to know how to run it.

**Folder structure explanation and challenges:**

Challenges:

There were not so many challenges to face while creating the project; the only challenge I faced was in between managing the re-arrangements of the columns. I was able to solve this challenge in the half hour to the time.

I had created a single function for all the fields, which was responsible for arranging all the rows. The text-formatted columns were able to be rearranged perfectly, but the numerical columns were complicated to rearrange. To solve this problem, I came to the conclusion that I need to have two functions, one for text-formatted columns and one for numerically formatted columns.

The functions for arranging the columns are:

SortData (for the text-formatted columns)

SortFloatData (for the numerically formatted columns)

**Creativity for Edit columns feature:**

The functions for editing the columns is : editcompleteRow.

Description: To update or edit the row, we can double-click on the existing row, and then it will redirect to the editing form. We can add details that we want to update or edit, and then we can click on the submit button, and it will redirect to the existing chemical supplies table where the updated or edited details would be shown.

**Known issues and credits:**

**Issues:**

If the internet connection is lost, then the font-awesome and bootstrap libraries do collapse because most of the event triggers are based on font-awesome library icons. Once the internet connection is resumed, the code will work fine.

**Credits:**

**[w3schools](https://www.w3schools.com/)**

**[fontawesome](https://fontawesome.com/)**

**[getbootstrap](https://getbootstrap.com/)**

**Software:**

Virtual Studio Code

Chrome or any other browser

**Notes on How to use :**

* To re-arrange the table either in ascending or descending order, please click on the respective column name.
* To add new entries, click on the + icon from the right side, add the details, and submit the form; your information will be shown in the table.
* To delete any entry, please select the row you want to delete and click on the dustbin icon from your right side.
* To refresh the table, please click on the rotate icon from the right side, and the table will get refreshed.
* To shift a row below any element or above any element, then select the row and then click on arrow up and arrow down from the right side as per your convenience.
* To edit entire row details, double-click on the row, and the form will appear with the existing data in the form. Change the data as per your convenience, and save the form. The data will be shown in the table.

**Precautions/Warnings:**

There are chances that any of the working functions can fail: Because there are multiple event listeners working simultaneously to make this app work. At that time, please refresh the tab or open the project in a new tab; it will reset all the event listeners, and functionality will work again.

In my code, I have tried to detach and reattach all the event listeners based on the work required, but the above point could be the worst solution.

I have tried handling all the event listeners to work smoothly together as per my knowledge.

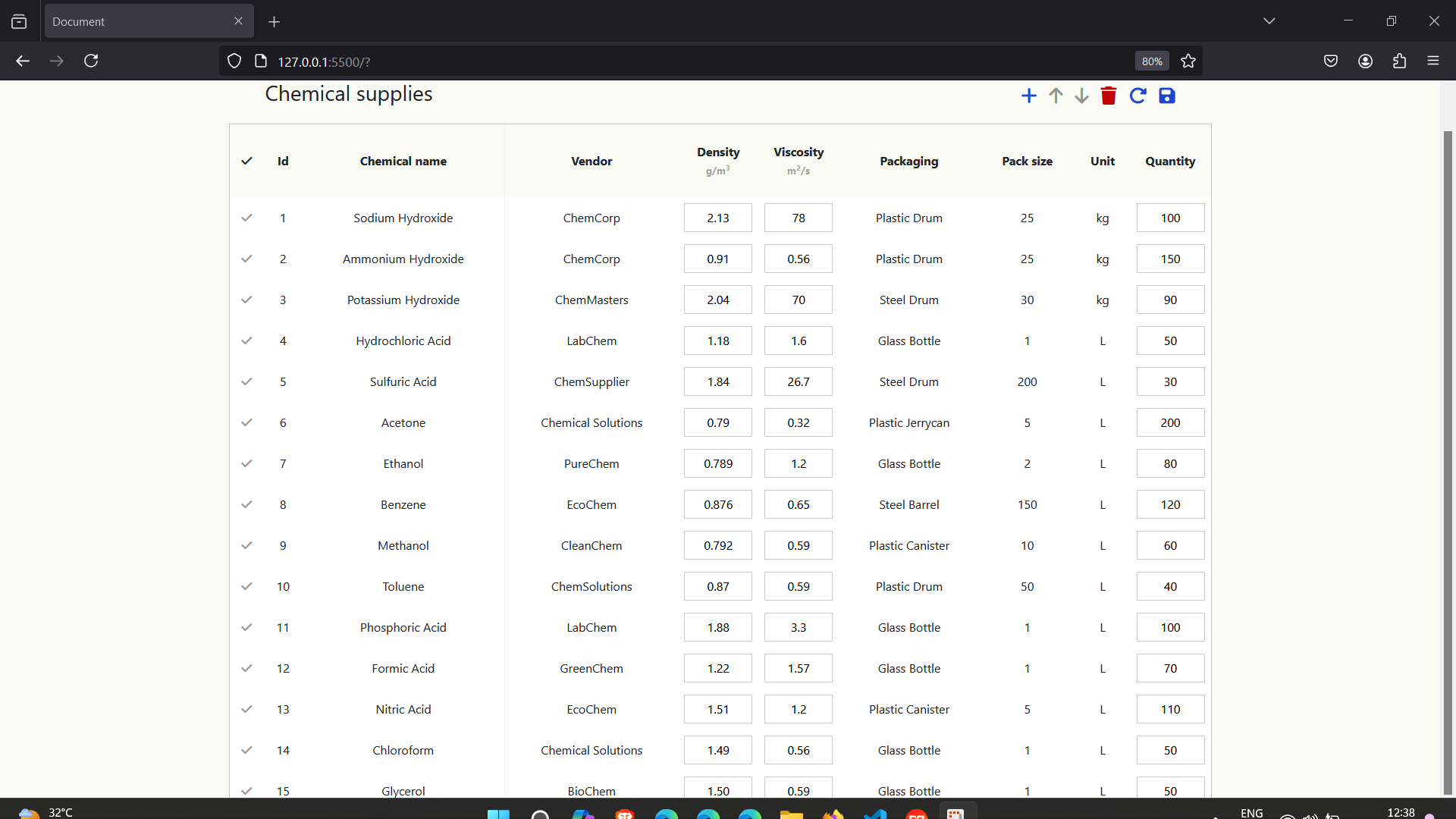
**Future Scope:**

Currently, there are no messages to notify the user that the changes that he/she tried to commit have been committed. In the future, we can add this function to make it more user-friendly.

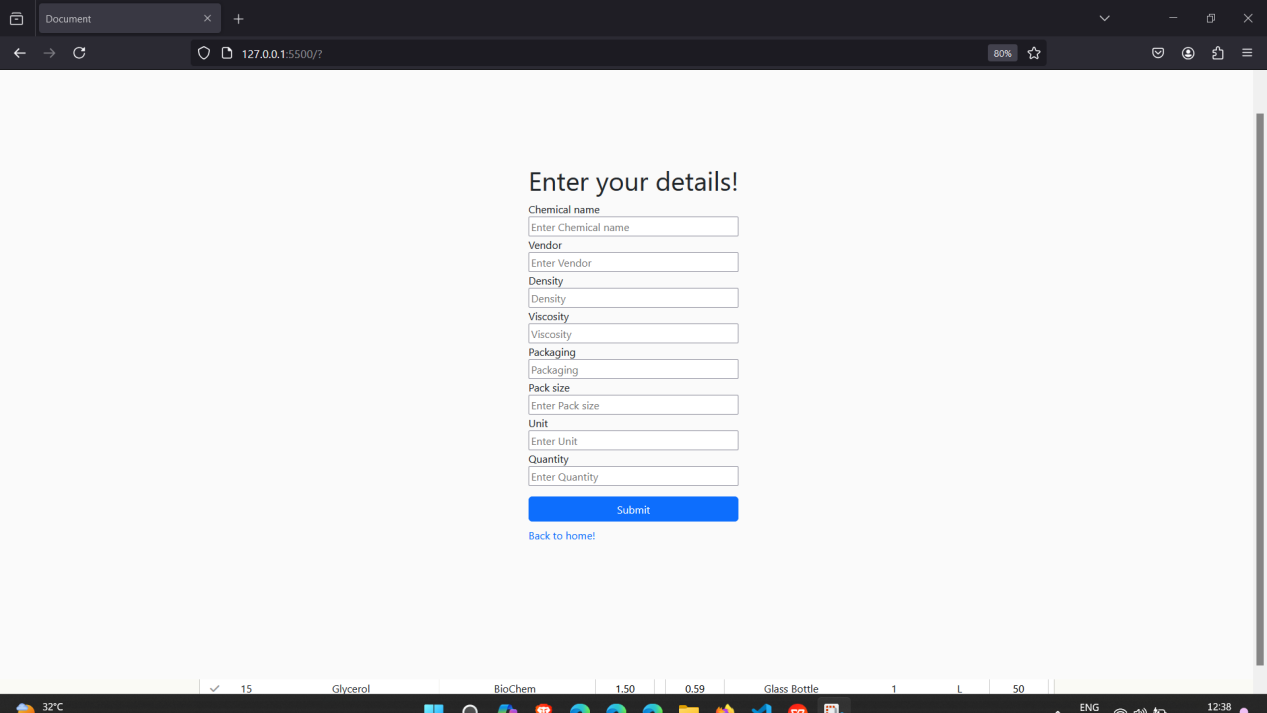
To make the functions more effective, we can handle all the event listeners/states in the framework like React and View using their state management libraries such as Redux or Vuex. With this, we can avoid the worst situation in the future.

**Output Screenshots:**

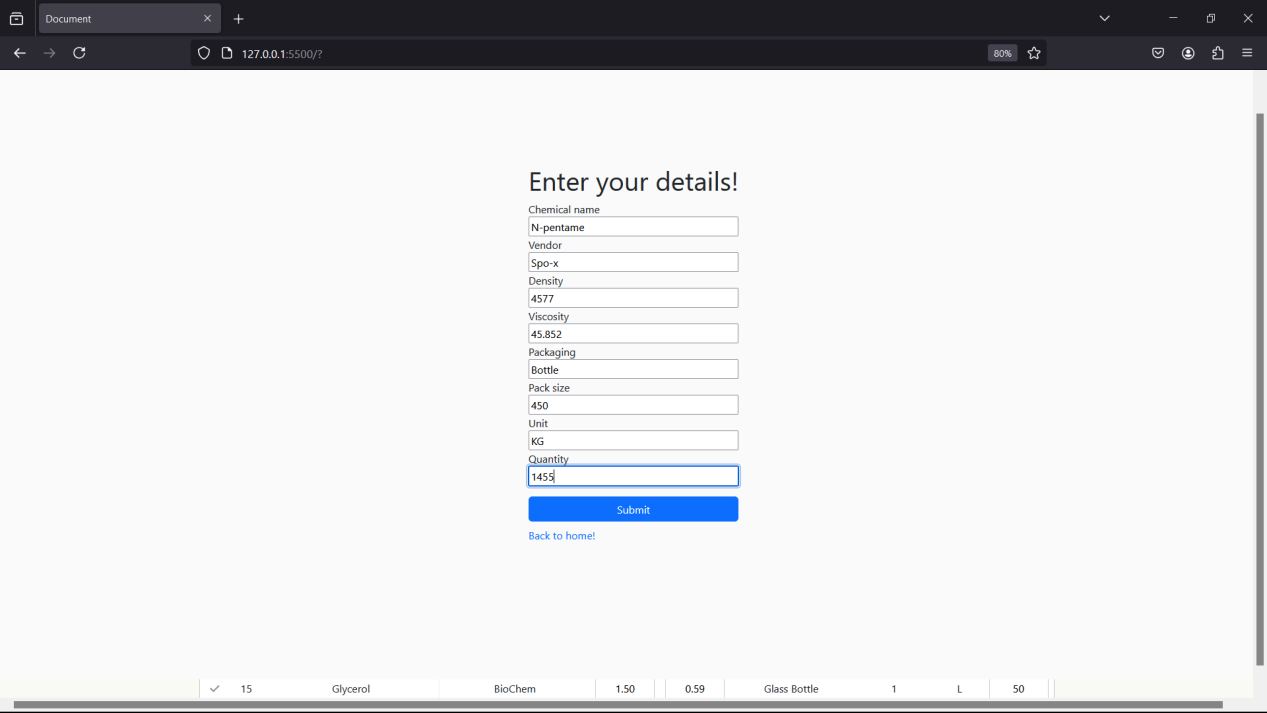
**Existing table (with 15 data rows):**

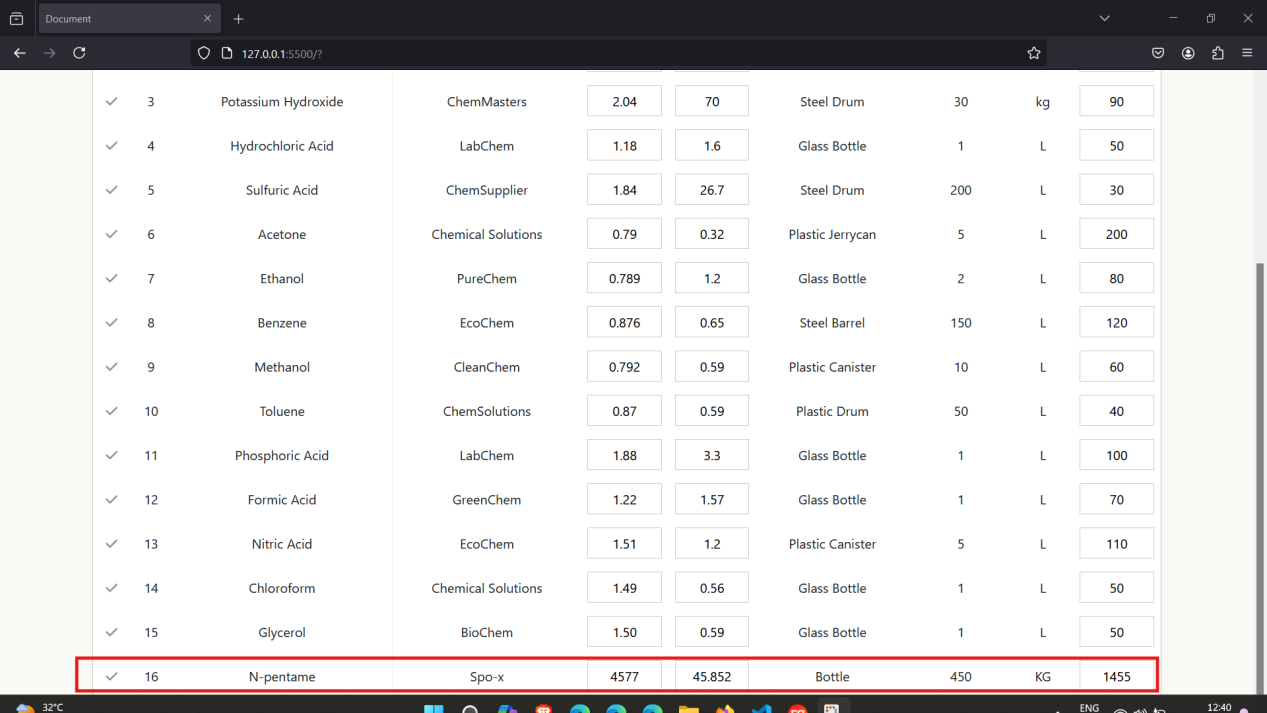


**Adding columns: Form before adding details.**



**After adding details:**

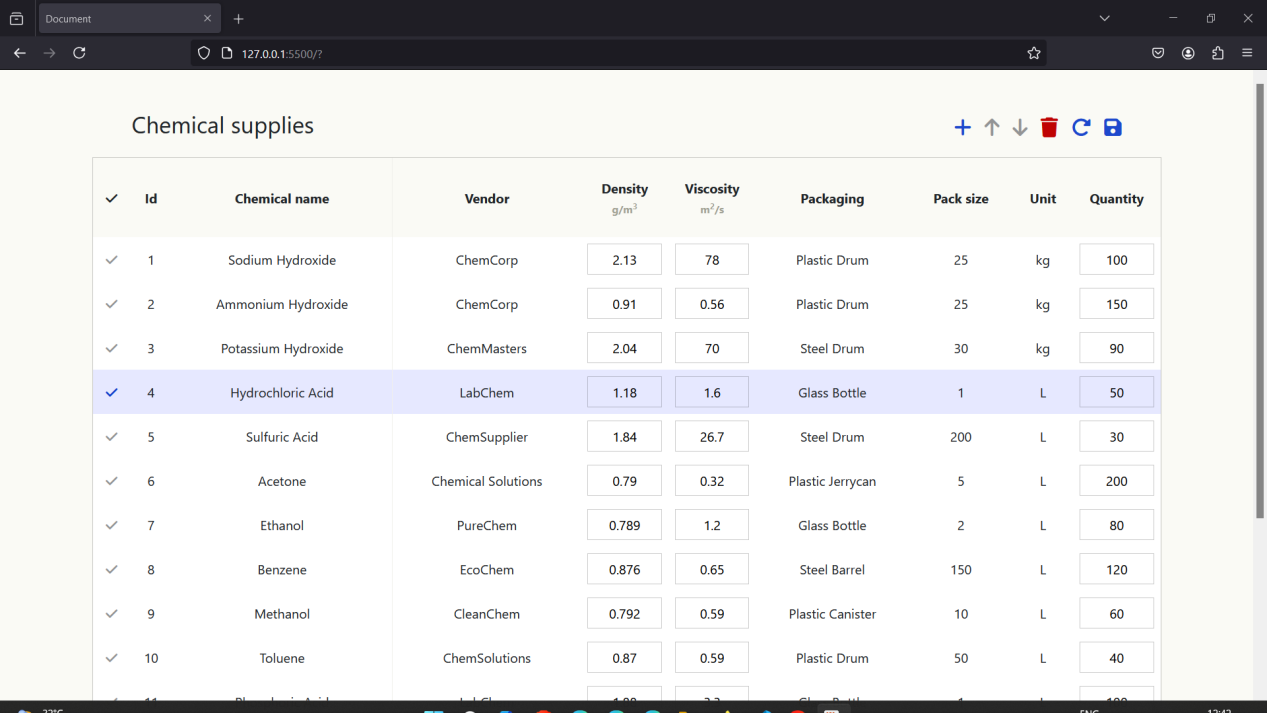




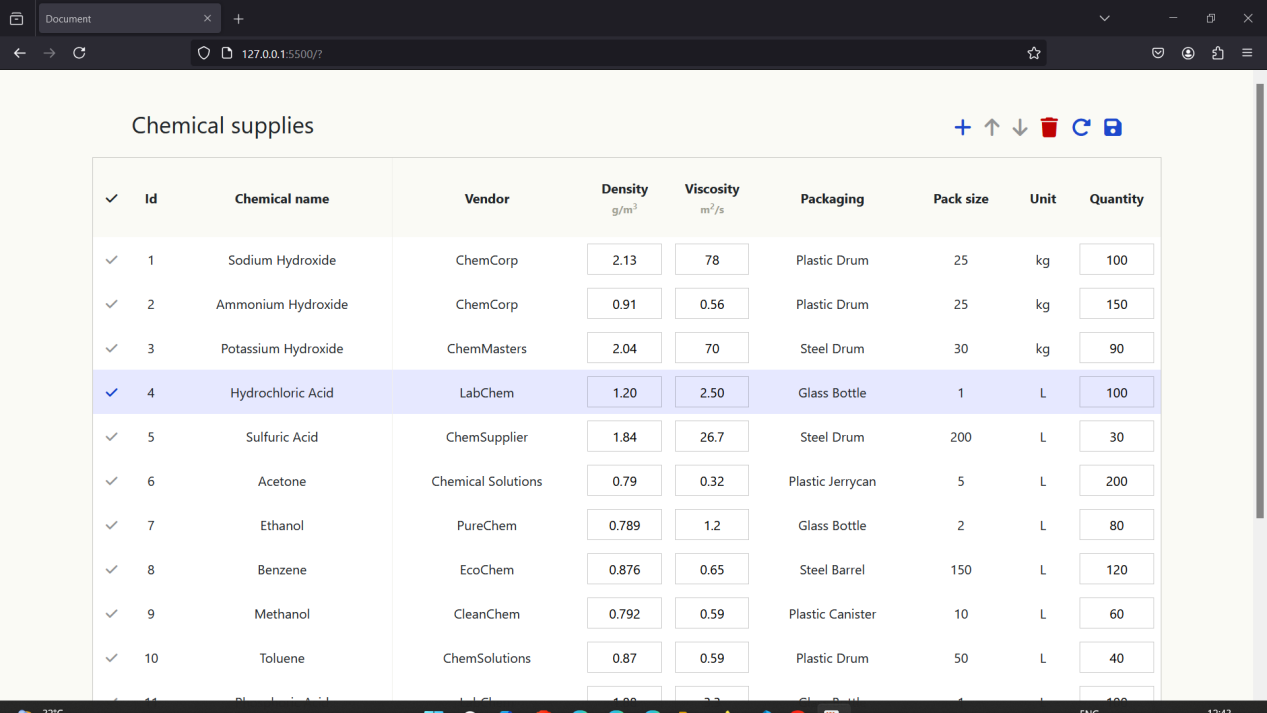
**Updating columns:**

**Updating the density, viscosity, and quantity columns:**

**Before updating row number 4 (editing it by adding new values to the text-boxes available in the table):**

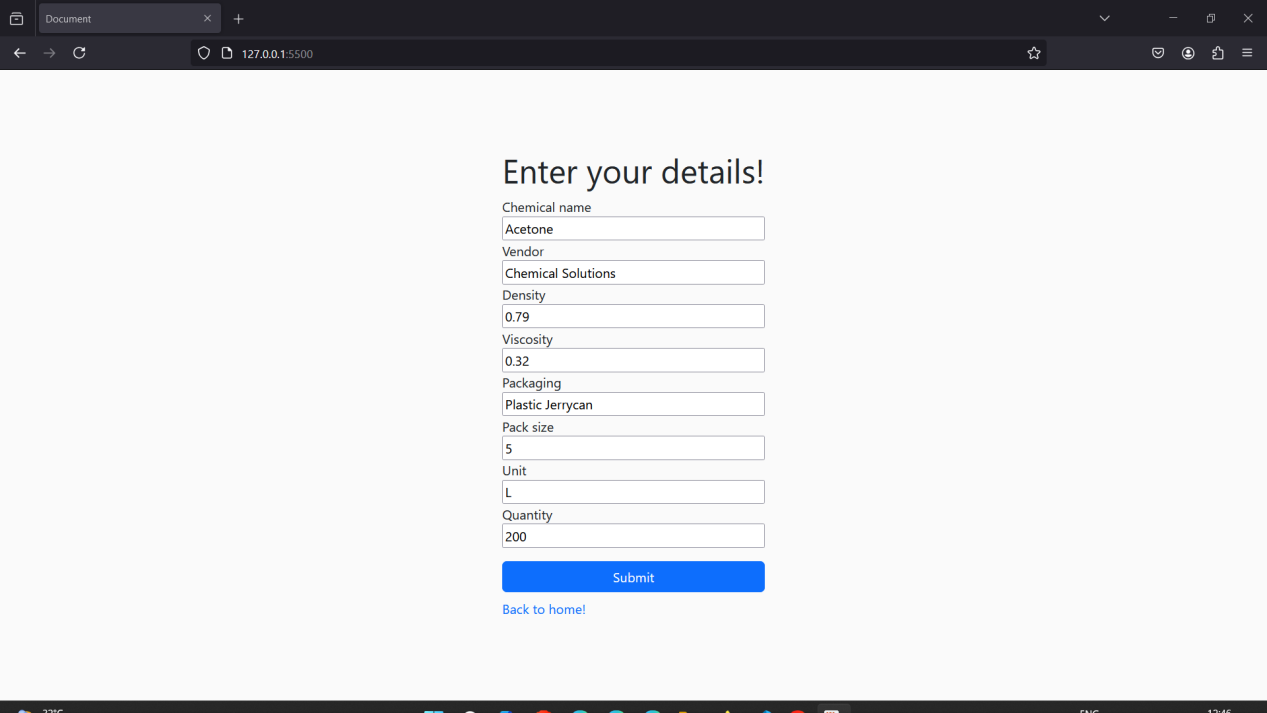


**After updating row number 4 (editing it by adding new values to the text-boxes available in the table):**

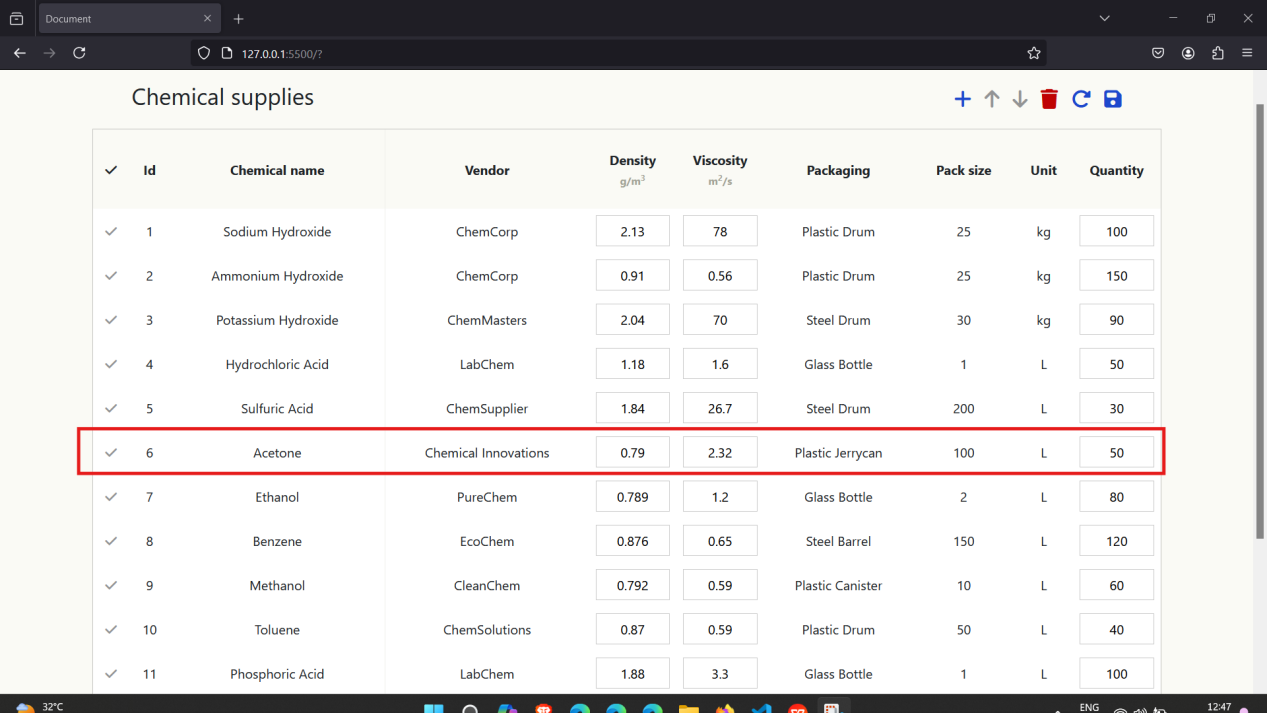


**Updating other columns:**

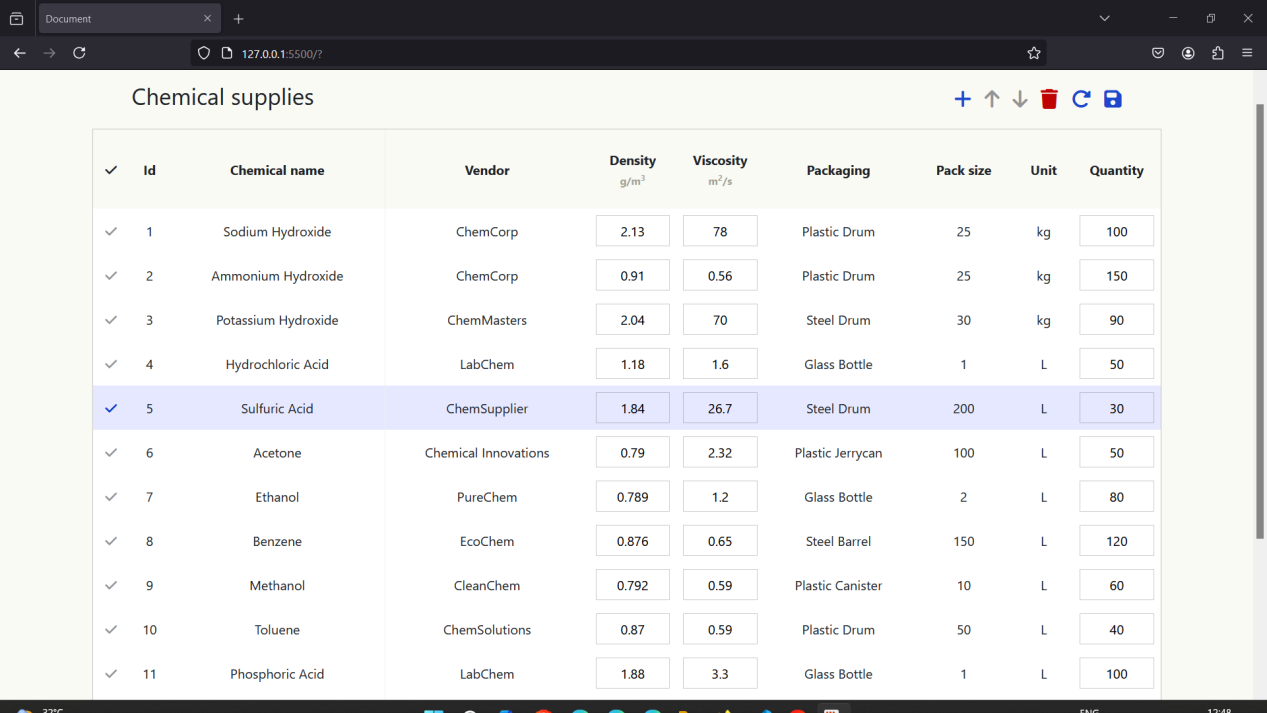
**Before updating other columns:**



**After updating other columns:**



**Deleting rows: Before Deleting the row number 5**



**After Deleting the row number 5:**

