1. Retrieve stock data from the database:

- Execute the query "SELECT date\_created, quantity FROM stock\_list" using `$conn->query()`.

- Initialize an empty array `$stock\_data`.

- Iterate over the result using a `while` loop:

- Fetch each row and add it to the `$stock\_data` array.

2. Retrieve purchase data from the database:

- Execute the query "SELECT item\_id, SUM(quantity) as total\_quantity FROM po\_items GROUP BY item\_id" using `$conn->query()`.

- Initialize an empty array `$purchase\_data`.

- Iterate over the result using a `while` loop:

- Fetch each row and add it to the `$purchase\_data` array.

3. Sort the stock data by quantity in descending order:

- Create a JavaScript variable `sortedStockData` and assign the JSON-encoded `$stock\_data` using `<?php echo json\_encode($stock\_data); ?>`.

- Sort the `sortedStockData` array using the `sort()` method with a comparison function that compares the `quantity` property of each object in descending order.

4. Find the dates with the highest, lowest, and mid-level stock:

- Assign the `date\_created` property of the first element in `sortedStockData` to the variable `highestStockDate`.

- Assign the `date\_created` property of the last element in `sortedStockData` to the variable `lowestStockDate`.

- Calculate the index of the mid-level stock by dividing the length of `sortedStockData` by 2 and rounding it to the nearest whole number. Assign the `date\_created` property of the element at the calculated index to the variable `midStockDate`.

5. Generate a list of stock suggestions:

- Get the reference to the HTML element with the ID "suggestions-list" and assign it to the variable `suggestionsList`.

- Clear the previous suggestions in the list by setting `suggestionsList.innerHTML` to an empty string.

- Create three list items (`<li>`) and append them to `suggestionsList`:

- Set the text content of `listItem1` to the suggestion for reducing stock on the highestStockDate.

- Set the text content of `listItem2` to the suggestion for ordering more stock on the lowestStockDate.

- Set the text content of `listItem3` to the suggestion for restocking on the midStockDate.

6. Add a purchase suggestion:

- Create a JavaScript variable `sortedPurchaseData` and assign the JSON-encoded `$purchase\_data` using `<?php echo json\_encode($purchase\_data); ?>`.

- Sort the `sortedPurchaseData` array by the `total\_quantity` property in descending order.

- Create a list item (`<li>`) `purchaseSuggestion` and set its text content to the suggestion for purchasing less of the item with the highest total quantity.