"*Sadguru's Amrit-Tulya - Tea Shop*" needs you to test their website that helps track the different items they have for sale.

The functional requirement includes Frontend and Backend specifications.

**Frontend Requirements:**

Create a page with a component at the top that can be used to add a new item to the inventory. The component should require a name, description, and price for the new item. You can upload a picture of the item as well. Below the component that adds new items to the inventory, display a list containing the name and price of each item in the inventory. If your app supports image uploads, you can display a thumbnail here as well. Each item in the list should have a button to remove it from the inventory. Clicking on an item should bring the user to a new page with all the information about the selected item (name, description, price, and optionally photo).

All API calls should be asynchronous and not cause any full-page reloads. The only full-page reloads the user should see are those caused by navigating between the two pages of the app

Regarding visual styles, the only requirement is that the content of both pages should be both horizontally and vertically centered.

**Backend Requirements:**

All of the functions of the frontend needs to be served by an API call and the store’s inventory should be persisted across restarts of the backend process. It will use a SQL database for any persistent data storage needs.

**Tracking:**

Please do your best to keep track of how long you spend on the functional analysis of each of the backend, frontend functionality, the frontend presentational aspects, and Test Case creation.

**QA Requirements:**

The deliverable shall be shared via **GitHub, BitBucket, SourceTree, or any versioning System**containing all relevant Test Cases in the correct sequence.

**Evaluation and Scoring:**

The design will be simple and minimal. You are free to ask any questions for clarification or help.

You will be evaluated on the below attributes -

- Test Case completeness of Frontend requirements

- Test Case completeness of Backend requirements

- Identify and define Test Cases for automation

Frontend Scenario: Checking the functionality of “Add new item” component.

1. Validate if the Component for “Add new item” to the inventory should be visible.
2. Validate if the Component: “Add new item” is clickable.
3. Validate if the mandatory Fields: Name, Description , Price are present under Component “Add new item”.
4. Validate the inventory list is displayed below the “Add new item” component.
5. Validate if the User is able to Add the item to the inventor by providing mandatory details.
6. Validate if the user is not able to Add the item if any item name field is empty.
7. Validate if the user is not able to Add the item if any item description field is empty.
8. Validate if the user is not able to Add the item if any item price field is empty.
9. Validate if the user is able to add an item by uploading a photo and mandatory details.
10. Validate if the user is able to remove an item from the inventory by clicking on the remove button.
11. Validate the page refresh is not observed while Add an item to the inventory.
12. Validate the page refresh is not observed while Removing an item from the inventory.
13. Validate if the user is redirected to a new page with a page refresh when clicked on the item hyperlink.
14. Validate name, description, price and photo (optional) are displayed when user clicks on an item.
15. Validate the page elements are vertically and horizontally aligned and centered.

Backend Scenario: Checking persistency of data added while adding a new item.

1. Validate if already added item in the inventory is present in the database.
2. Validate if the data of an item is persistent once the item is added to the inventory.

Automation Scenarios.

1. Validating UI correctness and presence of Add , Remove and List of item on the UI.
2. Validating addition and removal of items from the inventory.
3. Validating page refresh occurs only when user navigates between the pages.