Scenario: Building a Restaurant Website

Step 1: Project Setup

- User Story: As a developer, I want to set up a new project folder to build a restaurant website.
- Task: Create a new folder for the project.
- User Story: As a developer, I want to create the essential files for the project.
- Task: Inside the project folder, create three files: `index.html`, `styles.css`, and `script.js`.

Step 2: HTML Structure

- User Story: As a restaurant owner, I want a homepage for my restaurant's website with the restaurant's name and a menu.
- Task: In `index.html`, create a header with the restaurant's name and a navigation menu.
- Task: Create a section for the menu items.

Step 3: Menu Structure

- User Story: As a customer, I want to see a list of menu items with names, descriptions, and prices.
- Task: Inside the menu section, create a list of menu items using HTML lists.
- Task: Each menu item should include a name, description, and price.

Step 4: CSS Styling

- User Story: As a designer, I want the website to be visually appealing with proper styling.
- Task: In `styles.css`, style the restaurant name, navigation menu, and menu items.
- Task: Use CSS3 to make the website visually appealing. Apply colors, fonts, and layout styles.
- Task: Add styling for buttons and the order form.

Step 5: Interactive Menu

- User Story: As a customer, I want to view more details about a menu item when I click on it.
- Task: In `script.js`, use JavaScript to make the menu items interactive.
- Task: When a user clicks on a menu item, show a popup with item details.
- Task: Allow users to add items to their order by clicking an "Add to Order" button.

Step 6: Order Form

- User Story: As a customer, I want to place an order by providing my name, table number, and special instructions.
- Task: Create a form in the order section for customers to enter their name, table number, and special instructions.
- Task: Use HTML5 input types and attributes for validation.
- Task: Add a "Submit Order" button.

Step 7: JavaScript Functionality

- User Story: As a customer, I want to ensure that I've entered valid information in the order form.
- Task: In `script.js`, handle form submission.
- Task: Validate the form inputs using JavaScript.
- Task: Calculate the total order cost based on the selected items.
- Task: Display a confirmation message to the user.

Step 8: Testing

- User Story: As a tester, I want to verify that the website works correctly on various web browsers.
- Task: Test the website in different web browsers to ensure compatibility.
- Task: Verify that the form validation and menu interactions work as expected.

Step 9: Responsive Design

- User Story: As a user, I want the website to look good on both desktop and mobile devices.
- Task: Add CSS3 media queries to make the website responsive for various screen sizes.
- Task: Ensure that the website layout adjusts appropriately for both desktop and mobile screens.

Step 10: Deployment

- User Story: As a developer, I want to make the website accessible online for customers.
- Task: Choose a web hosting service to deploy your restaurant website.
- Task: Upload the HTML, CSS, and JavaScript files to the hosting server.
- Task: Ensure that the website is accessible online.

Step 11: Maintenance

- User Story: As a restaurant owner, I want to keep the website up to date and functional.
- Task: Regularly update the menu and content as needed.
- Task: Monitor the website for any issues and fix them promptly.
- Task: Gather feedback from customers and make improvements accordingly.

This detailed breakdown of the case study includes user stories and specific tasks for each step in building the restaurant website using HTML5, CSS3, and JavaScript. It emphasizes the different roles involved in the project, from developers and designers to customers and testers.

TIPS to Upload code on Github

Step 1: Create a GitHub Repository:

- Go to GitHub and log in to your account.
- Click the "+" sign in the top right corner and select "New repository."
- Fill in the repository name, description, and other settings. Make sure to choose whether your repository should be public or private.
- Click "Create repository."

Step 2: Initialize a Local Git Repository:

- Open your terminal.
- Navigate to the root directory of your project.
- Run the following commands to initialize a local Git repository and set up your project for version control.

```
git init
git add .
git commit -m "Initial commit"
```

Connect Your Local Repository to GitHub:

- In your GitHub repository, you should see the "Quick setup" section. It provides the URL you need for the next step. It will look something like this:

```
git remote add origin https://github.com/ParthShuklaa/SLKDET2023.git
```

- Replace the URL with the one provided for your repository.

Push Your Code to GitHub:

- Use the following command to push your code to GitHub:

```
git push -u origin master
```

If you're working with a different branch, replace "master" with the name of your branch.

Authentication and Credentials:

- When you run the `git push` command, Git might prompt you to enter your GitHub credentials if you haven't already configured Git to remember your credentials. You can configure this using Git's credential helper or by using a personal access token for authentication.

To configure the credential helper, run:

```
git config credential.helper store
```

This will store your credentials on your local machine temporarily.

Verify Your Code on GitHub:

- Go to your GitHub repository in your web browser, and you should see your code and files there.

Your code is now uploaded to GitHub, and you can continue to push updates and changes to your repository using the same Git commands (`git add`, `git commit`, and `git push`). Make sure to update your code and commit changes before pushing them to your repository.