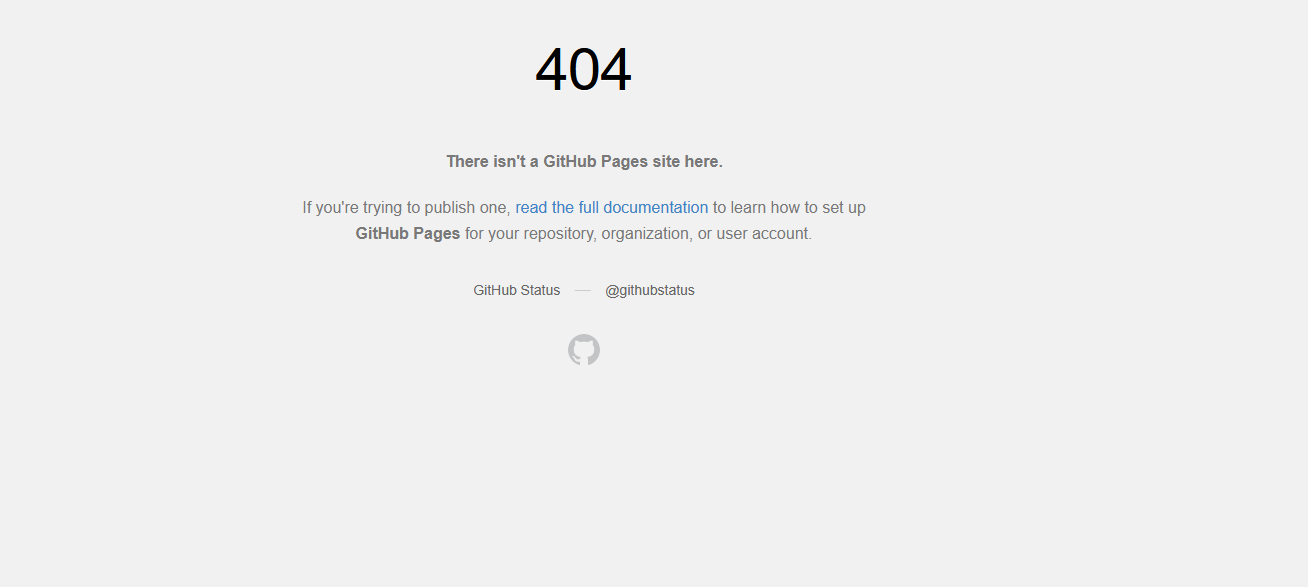
| **Project:** Test Plan |  |
| --- | --- |
| **Course: CSD 280 DevOps** |  |
| **Description: 5 test plans for the Todo Application.** |  |
| **Date: 2025/04/13** |  |

Robert Dylan Bean

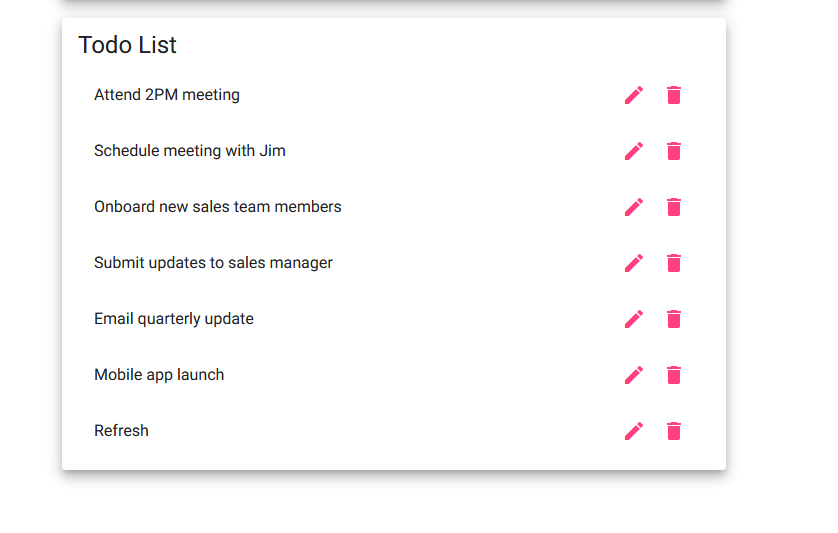
4/13/2025

Professor John Woods

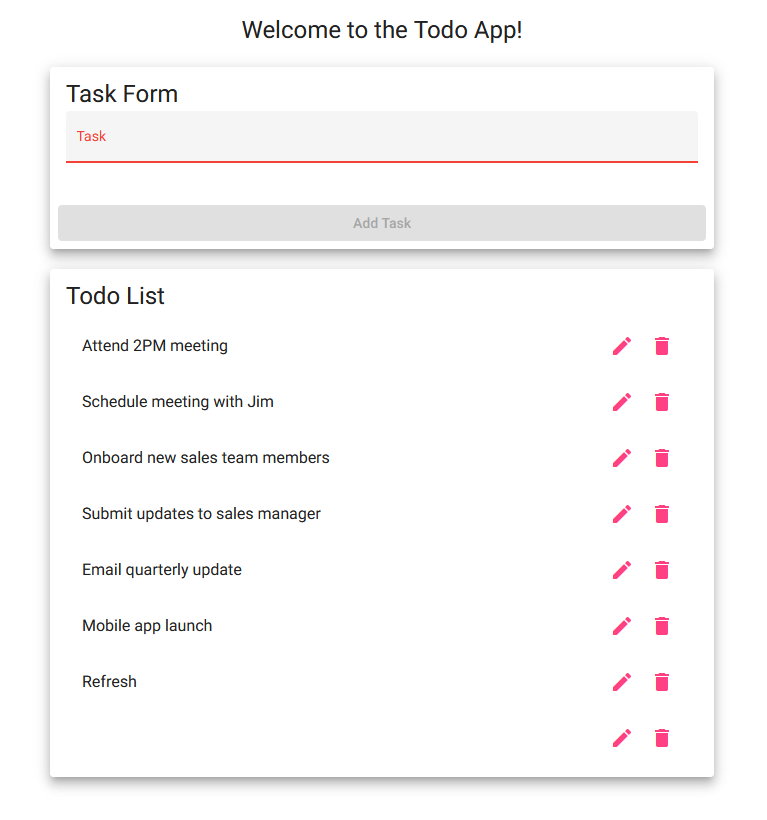
Assignment 4.2

| Test 1 | **Verify that navigating to an invalid URL within the Todo application displays a custom 404 error page.** | | | |
| --- | --- | --- | --- | --- |
|  | **Test Objective:** Ensure that when users enter an incorrect or non-existent path within the application, a custom 404 error message is presented instead of a default server error. | **Developer:** Robert Bean  **Date tested:** 2025/04/13 | **Peer tester: William Renard**  **Date tested:** 2025/04/13 | |
| **Step** | **Action** | **Expected results:** | **Developer pass/fail** | **Tester pass/fail**  **+ Screenshot** |
| 1 | Open a supported web browser. | Browser is launched successfully. | yes | yes |
| 2 | Enter an invalid URL in the address bar, e.g., <https://buwebdev.github.io/todo/invalidpage> | The page reloads correctly with all elements displayed. | yes | yes |
| 3 | Refresh the invalid URL page. | The custom 404 error page reloads correctly and continues to display the error message. | yes | yes |
| **Comments** | Entering invalid url like as seen in step 2 will take you to 404 page. Refreshing the invalid page stays on the 404 page. | | | |

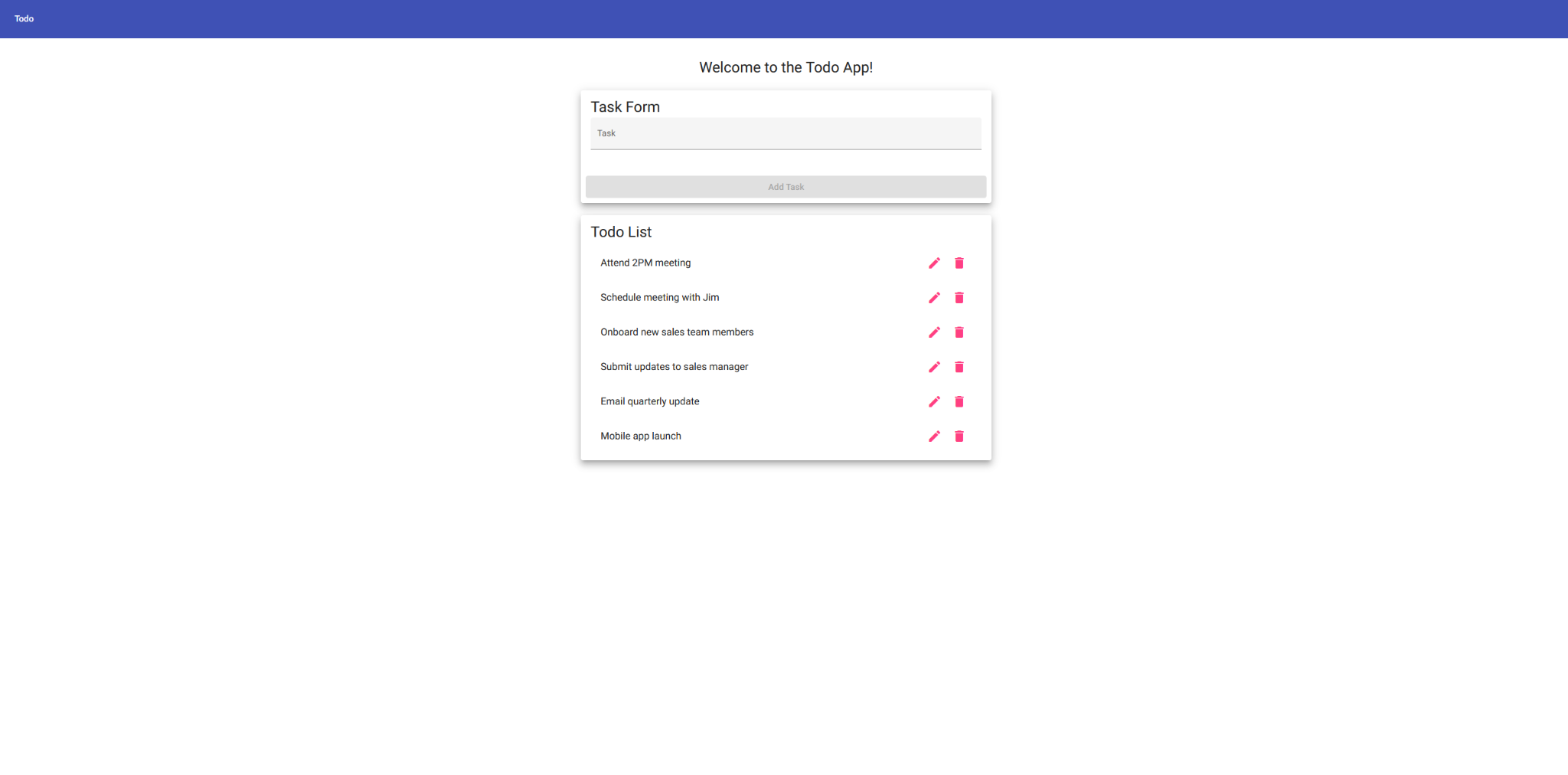
| Test 2 | **Validate that a new todo task item can be created successfully.** | | | |
| --- | --- | --- | --- | --- |
|  | **Test Objective:** To ensure that the “Add Task” functionality meets the requirement for users to add a new todo task item. | **Developer:** Robert Bean  **Date tested:** 2025/04/13 | **Peer tester: William Renard**  **Date tested:**2025/04/13 | |
| **Step** | **Action** | **Expected results:** | **Developer pass/fail** | **Tester pass/fail** |
| 1 | Locate and click on the text field titled “Task” | The text field has an insertion point displayed | yes | yes |
| 2 | Enter valid text into the task input field | The text is entered and visible in the input field. | yes | yes |
| 3 | Click the “Add Task” button. | The new todo task item appears in the task list. | yes | yes |
| 4 | Verify that the todo item persists after refreshing the page. | The newly added item remains visible in the task list. | no | no |
| **Comments** | ToDo list allows you to add new task. New task does not persist after refreshing the page. | | | |



| Test 3 | **Ensure that users cannot submit a new todo task item if the input field is left empty.** | | | |
| --- | --- | --- | --- | --- |
|  | **Test Objective:** To validate that the form enforces required fields. | **Developer:** Robert Bean  **Date tested:** 2025/04/13 | **Peer tester: William Renard**  **Date tested:** 2025/04/13 | |
| **Step** | **Action** | **Expected results:** | **Developer pass/fail** | **Tester pass/fail** |
| 1 | Locate and click on the text field titled “Task” | The text field has an insertion point displayed | yes | yes |
| 2 | Leave the todo input field blank and click the “Add Task” button. | The system prevents submission | yes | yes |
| 3 | Enter whitespace in the todo input field and click “Add Task.” | The system again prevents submission | no | no |
| **Comments** | So apparently if you add more than 5 whitespace characters it does in fact allow you to submit the task. This one is a fail, if you add just one or two whitespace characters the button remains inactive. | | | |



| Test 4 | **Verify that users can successfully edit an existing todo task item and that empty edits are rejected.** | | | |
| --- | --- | --- | --- | --- |
|  | **Test Objective:** To ensure that the edit functionality works as intended and validates input. | **Developer:** Robert Bean  **Date tested:** 2025/04/13 | **Peer tester: William Renard**  **Date tested:** 2025/04/13 | |
| **Step** | **Action** | **Expected results:** | **Developer pass/fail** | **Tester pass/fail** |
| 1 | From the todo list, select a task item to edit (click on the “Edit” button). | The edit form or input field is pre-populated with the current task details. | yes | yes |
| 2 | Modify the text to a new valid task description | The input field reflects the updated text. | yes | yes |
| 3 | Click the “Save” button. | The updated task description appears in the task list. | yes | yes |
| 4 | Attempt to clear the text completely and click “Save” to check validation. | The system prevents an empty submission | no | no |
| 5 | Refresh the page to ensure that changes persist. | The updated task remains correctly displayed after a page refresh. | no | no |
| **Comments** | Refreshing the page even after doing steps 2-4 does not persist. Step 4 also does not allow you to do an empty submission | | | |



| Test 5 | **Verify that users can delete an existing todo task item.** | | | |
| --- | --- | --- | --- | --- |
|  | **Test Objective: T**o ensure that the “Delete” functionality works and that the item is removed from the list. | **Developer:** Robert Bean  **Date tested:** 2025/04/13 | **Peer tester: William Renard**  **Date tested:** 2025/04/13 | |
| **Step** | **Action** | **Expected results:** | **Developer pass/fail** | **Tester pass/fail** |
| 1 | Identify a todo task item in the list intended for deletion. | The item is clearly marked and selectable for deletion. | yes | yes |
| 2 | Click on the “Delete” button associated with the selected task. | The todo task item is removed from the list. | yes | yes |
| 3 | Refresh the page to ensure that the deleted item does not reappear. | The task item remains removed from the list after refresh. | no | no |
| **Comments** | Again like in previous cases, state does not persist on refresh. However, delete functionality does exist and work properly. | | | |

