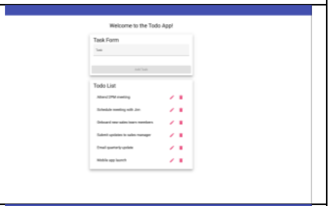
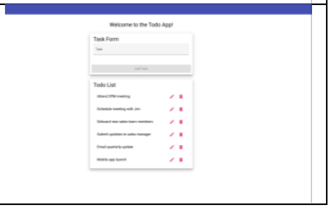
Todo Application Functional Test Plan

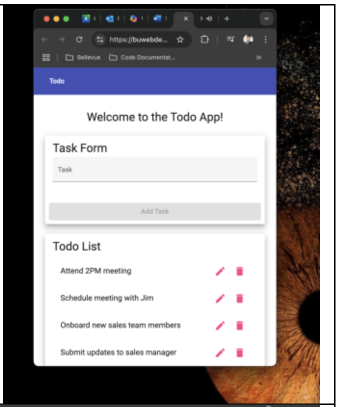
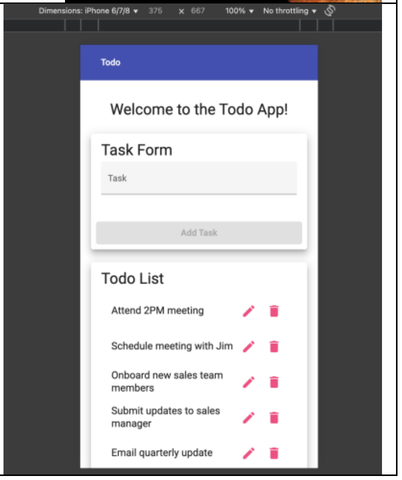
Project: Todo Application	**Developer:** Tazmin Somerville
Course: CSD 380 - DevOps	
Description: Test Plan for verifying core functionalities of the Todo Application	
Date: 2025/04/09	

TABLE OF CONTENTS

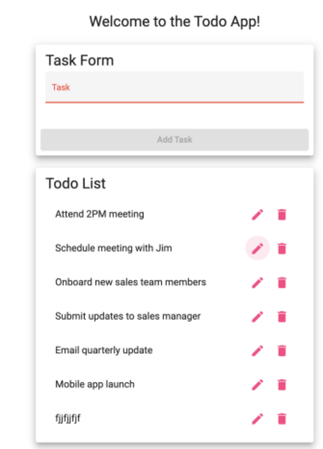
Todo Application Functional Test Plan

Test 1	**Verify Landing Page Loads (Requirement #10 + Prototype Check)**			
	Test Objective: Ensure that users can access the Todo website through its standard URL, and that the landing page loads correctly without errors.	**Developer:** Tazmin Somerville **Date tested:** 2025/04/09	**Peer tester:** **Noel Miranda** **Date tested:** 2025/04/10	
Step	**Action**	**Expected results:**	**Developer pass/fail**	**Tester pass/fail** **+ Screenshot**
1	Visit https://buwebdev.github.io/todo/	Landing page loads without errors	<yes/no> - Yes	Pass  Welcome to the Todo App! Task Form Task Add Task Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch
2	Refresh the browser window	The page reloads correctly without errors	<yes/no> - Yes	Pass  Welcome to the Todo App! Task Form Task Add Task Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch

Todo Application Functional Test Plan

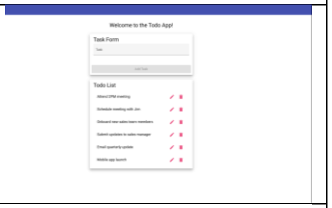
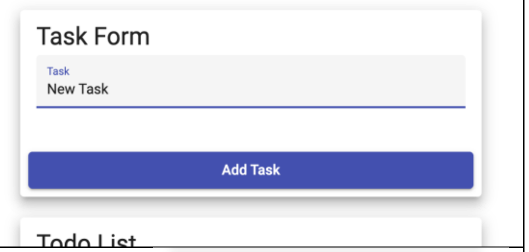
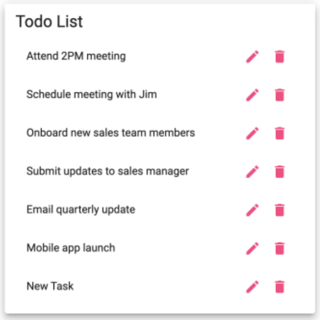
3	Inspect responsiveness on desktop browser by resizing window	Layout adjusts without breaking	<yes/no> - Yes	Pass	 Todo Welcome to the Todo App! Task Form Task Add Task Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager
4	Open the landing page on a mobile browser	Layout is responsive and readable	<yes/no> - Yes	Pass	 Dimensions: iPhone 6/7/8 375 × 667 100% No throttling Todo Welcome to the Todo App! Task Form Task Add Task Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update

Todo Application Functional Test Plan

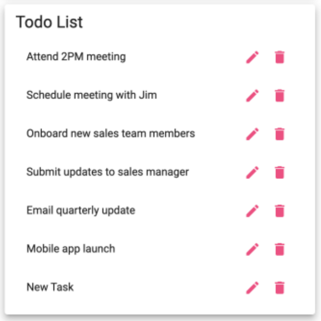
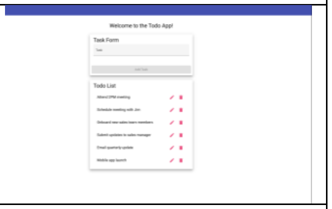
5	Use keyboard navigation (tab, enter) on input and buttons	Elements are accessible via keyboard	<yes/no> - Yes	Pass  Welcome to the Todo App! Task Form Task Add Task Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch fjjfjjfjf
Comments	This test case passed! The landing page loaded promptly with no noticeable errors or lag. Reloading the page through a browser refresh maintained the same smooth performance. All tested views displayed correctly without any visual issues. The only recommendation I could make is possibly not extending the list container so wide when resizing the browser to a certain dimension.			

Test 2	**Create a New Todo Task Item (Requirement #2)**

Todo Application Functional Test Plan

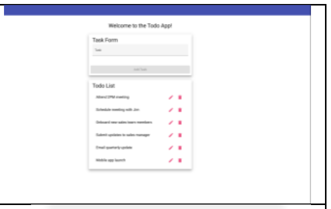
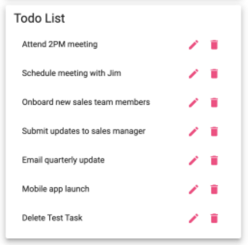
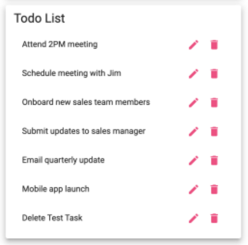
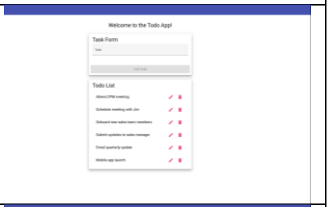
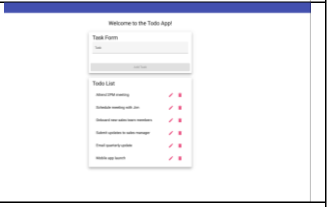
	Test Objective: Verify that users can add a new todo item via the input field on the landing page.	**Developer:** Tazmin Somerville **Date tested:** 2025/04/09	**Peer tester:** **Noel Miranda** **Date tested:** 2025/04/10	
Step	**Action**	**Expected results:**	**Developer pass/fail**	**Tester pass/fail**
1	Visit https://buwebdev.github.io/todo/	Landing page loads with visible input field	<yes/no> - Yes	Pass  Welcome to the Todo App! Task Form Task Add Task Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch
2	Click into the input field and type “New Task”	The input is accepted without errors	<yes/no> - Yes	Pass  Task Form Task New Task Add Task Todo List
3	Click the **Add Task** button	The new task is added to the list below	<yes/no> - Yes	Pass  Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch New Task

Todo Application Functional Test Plan

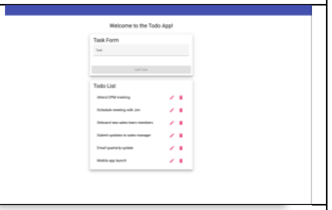
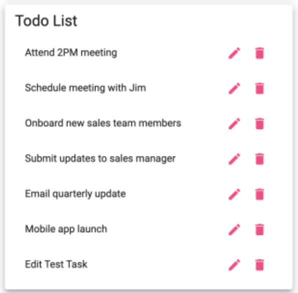
4	Confirm the new task item displays with an edit icon and delete icon	Item is added correctly with full functionality	<yes/no> - Yes	Pass  Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch New Task
5	Refresh the browser window	Task is cleared after refresh; persistence not required by this test case	<yes/no> - Yes	Pass  Welcome to the Todo App! Task Form Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch
Comments	This test case passed! The task was added to the list through the input field and the Add Task button without any issues. The interface appeared user-friendly and responded well to interaction. Refreshing the browser resulted in the task disappearing, which suggests that the current version of the application does not include data persistence through local storage or a connected backend. Other than that, the application meets the business requirement. However, since test 1 already covers the landing page, I suggest removing step 1 from this test case to avoid redundancy. It may be more appropriate to list it as a precondition instead.			

Test 3	**Delete a Todo Task Item (Requirement #4)**			
	Test Objective: Verify that users can delete an existing task from the todo list.	**Developer:** Tazmin Somerville **Date tested:** 2025/04/09	**Peer tester: Noel Miranda** **Date tested:** 2025/04/10	
Step	**Action**	**Expected results:**	**Developer pass/fail**	**Tester pass/fail**

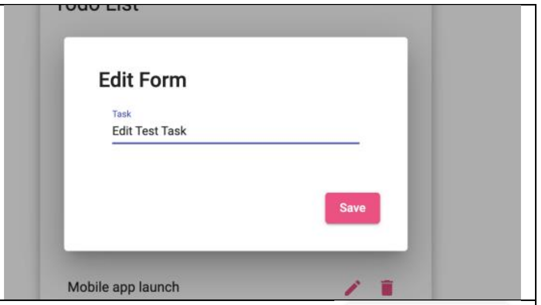
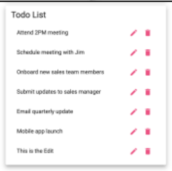
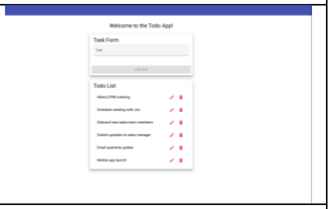
Todo Application Functional Test Plan

1	Visit https://buwebdev.github.io/todo/	Page loads with visible input and existing tasks	<yes/no> - Yes	Pass	 Welcome to the Todo App! Task Form Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch
2	Add a new task named “Delete Test Task”	Task is added to the list below	<yes/no> - Yes	Pass	 Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch Delete Test Task
3	Locate the “Delete” (trash can) icon next to the new task	Delete icons are visible and clickable	<yes/no> - Yes	Pass	 Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch Delete Test Task
4	Click the “Delete” icon	Task is immediately removed from the list	<yes/no> - Yes	Pass	 Welcome to the Todo App! Task Form Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch
5	Confirm the task does not reappear on refresh	Task is gone permanently after refresh	<yes/no> - Yes	Pass	 Welcome to the Todo App! Task Form Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch
Comments	Test case passed! Well-constructed test case as well. Overall, Task deletion functioned as intended. The delete icon responded promptly and removed the task upon interaction. After refreshing the page, the task remained deleted, confirming that it did not				

	persist unexpectedly. However, I suggest potentially implementing a pop-up to prompt the user to confirm if they are sure they want to delete a Todo item before proceeding. This could lead to unintentional mistakes, thus impacting the user experience.

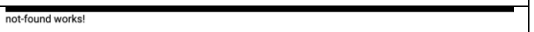
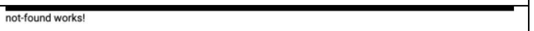
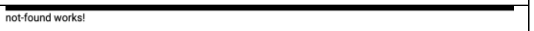
Test 4	**Edit a Todo Task Item (Requirement #3)**			
	Test Objective: Verify that users can edit an existing task on the list using the edit icon and update the task text.	**Developer:** Tazmin Somerville **Date tested:** 2025/04/09	**Peer tester: Noel Miranda** **Date tested:** 2025/04/10	
Step	**Action**	**Expected results:**	**Developer pass/fail**	**Tester pass/fail**
1	Visit https://buwebdev.github.io/todo/	Landing page loads with visible tasks	<yes/no> - Yes	Pass  Welcome to the Todo App! Task Form Task Add Task Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch
2	Add a task labeled “Edit Test Task”	Task is added to the list	<yes/no> - Yes	Pass  Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch Edit Test Task

Todo Application Functional Test Plan

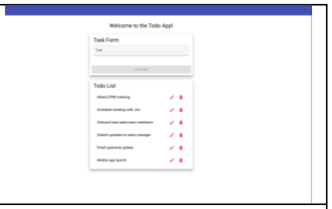
3	Click the “Edit” (pencil) icon next to the new task	Task text becomes editable	<yes/no> - Yes	Pass  Edit Form Task Edit Test Task Save Mobile app launch
4	Change the task text to 'This is the Edit' and click the Save button	Task is updated in the list with new text	<yes/no> - Yes	Pass  Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch This is the Edit
5	Refresh the page	Edited task disappears (no persistence)	<yes/no> - Yes	Pass  Welcome to the Todo App! Task Form Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch
Comments	Test case passed! The edit option operated without any issues. Clicking the pencil icon opened a dialog box where changes could be made, and those changes appeared right after selecting the save button. Similar to adding a task, edits were not retained after refreshing the browser, which matches how the application is presently built. The overall interaction felt straightforward and easy to follow.			

Test 5	**Invalid URL Handling (Requirement #5: Custom 404 Page)**

Todo Application Functional Test Plan

	Test Objective: Verify that users are routed to a basic 404-style message when attempting to access a non-existent URL.	**Developer:** Tazmin Somerville **Date tested:** 2025/04/09	**Peer tester: Noel Miranda** **Date tested:** 2025/04/10	
Step	**Action**	**Expected results:**	**Developer pass/fail**	**Tester pass/fail**
1	Navigate to an invalid URL (e.g., https://buwebdev.github.io/todo/invalid	User is redirected to /not-found route, which displays a 404-style message	<yes/no> - Yes	Pass  not-found works!
2	Observe the message on the /not-found page	Text "not-found works!" is displayed	<yes/no> - Yes	Pass  not-found works!
3	Confirm the page lacks advanced styling or navigation	The page is plain, with no back or home link	<yes/no> - Yes	Pass  not-found works!

Todo Application Functional Test Plan

4	Use the browser back button	User returns to the homepage (/todo) successfully	<yes/no> - Yes	Pass  Welcome to the Todo App! Task Form Task Add Task Todo List Attend 2PM meeting Schedule meeting with Jim Onboard new sales team members Submit updates to sales manager Email quarterly update Mobile app launch
Comments	When an invalid URL is accessed, the application navigates to a specific /not-found route. The displayed message confirms that the custom 404 route is functioning correctly, although the page itself is plain and lacks visual design elements or navigation options. There is no button to return to the main interface, but the browser's back feature continues to operate as expected. The behavior fulfills the requirement for a functioning 404 route within a single-page application. This test case matches the expected outcome and aligns with the business requirement. However, the user experience could be improved by centering the message, adding visual enhancements such as color, and including a short, polite note or an illustrative image to acknowledge the error more gracefully.			