

Requirement:		Requirement Document:				
1	Integrate Add Task with the SQLite for storage of New Tasks	<p>The diagram illustrates the 'Add Task' process. A User actor interacts with the 'Add Task' use case. This use case includes 'Enter Task Title', 'Enter Task Time', 'Enter Event Info', 'Enter Notes', 'Enter Category', and 'Enter Priority'. 'Enter Task Title' includes 'Add New Task' and 'Display Task Title Box'. 'Enter Task Time' includes 'Display Cancel Button' and 'Display Date and Time Picker'. 'Enter Event Info' includes 'Display Save Button', 'Display Category Picker', and 'Display Notes Box'. 'Enter Notes' includes 'Display modal to add new task'. 'Enter Category' includes 'Choose to Save Task' and 'Return To Calendar'. 'Enter Priority' includes 'Pass Task Info to Database'. 'Add Task' includes 'Choose to Cancel Adding New Task' and 'Save Task'. 'Choose to Cancel Adding New Task' includes 'Pass Task Info to Database'. 'Save Task' includes 'Pass Task Info to Database'. The 'Pass Task Info to Database' use case is associated with the 'Database' actor. The 'AddTaskModal Class' is associated with 'Display modal to add new task'. The 'Database Class' is associated with 'Pass Task Info to Database'.</p>				
2	Integrate Add Events with the SQLite for storage of New Events	<p>The diagram illustrates the 'Add Event' process. A User actor interacts with the 'Add Event' use case. This use case includes 'Enter Event Name', 'Enter Event Time', 'Enter Event Info', 'Enter Event Date', 'Choose to Save Event', 'Choose to Cancel Adding New Event', and 'Return To Calendar'. 'Enter Event Name' includes 'Add New Event' and 'Display Event Name Box'. 'Enter Event Time' includes 'Display Cancel Button' and 'Display Date and Time Picker'. 'Enter Event Info' includes 'Display Save Button' and 'Display modal to add new event'. 'Enter Event Date' includes 'Display modal to add new event'. 'Choose to Save Event' includes 'Return To Calendar'. 'Choose to Cancel Adding New Event' includes 'Pass Event Info to Database'. 'Return To Calendar' includes 'Pass Event Info to Database'. 'Pass Event Info to Database' includes 'Save Event'. The 'Pass Event Info to Database' use case is associated with the 'Database' actor. The 'AddEventModal Class' is associated with 'Display modal to add new event'. The 'Database Class' is associated with 'Pass Event Info to Database'.</p>				
3	Integrate so that the To-do list page pulls data using SQLite and populates the page	Feature	Description	UI Elements	Database Operation	Testing
		Load Tasks from SQLite	Display all tasks on the To-do list page by fetching from the database.	Task list area	get_all_tasks() retrieves all records	Confirm tasks load properly and display upon opening the To-do list page.
		Populate Task Details	Populate task UI components with data from the database (title, due date, priority, category).	Labels for each task attribute	Data mapping to UI elements	Verify each task item displays correct details from the database.
		Live Update on Add	Immediately show new tasks in the UI when added to the database.	Task list view	Auto-refresh task list after new entry	Ensure new tasks appear on screen right after addition.

Requirement:		Requirement Document:																																			
4	Integrate so that the Calendar view pulls data using SQLite and populates the page	+	<table><tr><th>Feature</th><th>Description</th><th>UI elements</th><th>Database operation</th><th>Testing</th></tr><tr><td>Load events from SQLite</td><td>Display all events on the calendar page by fetching from the database.</td><td>Calendar grid or list view</td><td><code>get_all_events()</code> retrieves all records</td><td>Confirm events load properly and display upon opening the calendar page.</td></tr><tr><td>Populate Event Details</td><td>Populate event UI components with data from the database (title, date, time, location, description)</td><td>Labels for each event attribute</td><td>Data mapping to UI elements</td><td>Verify each event item displays correct details from the database.</td></tr><tr><td>Live Update on add</td><td>Immediately show new events in the UI when added to the database</td><td>Calendar view</td><td>Auto-refresh calendar after new entry</td><td>Ensure new events appear on screen right after addition</td></tr><tr><td>Daily/Weekly view</td><td>Allow switching between daily, weekly, and monthly calendar views</td><td>View switcher</td><td>Change display mode for events</td><td>Confirm that switching between views adjusts the display as expected.</td></tr><tr><td>Highlight upcoming</td><td>Highlight events happening within a specified period (e.g., within the week)</td><td>Calendar view</td><td>Visual cues for upcoming events</td><td>Verify that events are highlighted correctly based on the time filter.</td></tr></table>					Feature	Description	UI elements	Database operation	Testing	Load events from SQLite	Display all events on the calendar page by fetching from the database.	Calendar grid or list view	<code>get_all_events()</code> retrieves all records	Confirm events load properly and display upon opening the calendar page.	Populate Event Details	Populate event UI components with data from the database (title, date, time, location, description)	Labels for each event attribute	Data mapping to UI elements	Verify each event item displays correct details from the database.	Live Update on add	Immediately show new events in the UI when added to the database	Calendar view	Auto-refresh calendar after new entry	Ensure new events appear on screen right after addition	Daily/Weekly view	Allow switching between daily, weekly, and monthly calendar views	View switcher	Change display mode for events	Confirm that switching between views adjusts the display as expected.	Highlight upcoming	Highlight events happening within a specified period (e.g., within the week)	Calendar view	Visual cues for upcoming events	Verify that events are highlighted correctly based on the time filter.
			Feature	Description	UI elements	Database operation	Testing																														
			Load events from SQLite	Display all events on the calendar page by fetching from the database.	Calendar grid or list view	<code>get_all_events()</code> retrieves all records	Confirm events load properly and display upon opening the calendar page.																														
			Populate Event Details	Populate event UI components with data from the database (title, date, time, location, description)	Labels for each event attribute	Data mapping to UI elements	Verify each event item displays correct details from the database.																														
			Live Update on add	Immediately show new events in the UI when added to the database	Calendar view	Auto-refresh calendar after new entry	Ensure new events appear on screen right after addition																														
			Daily/Weekly view	Allow switching between daily, weekly, and monthly calendar views	View switcher	Change display mode for events	Confirm that switching between views adjusts the display as expected.																														
			Highlight upcoming	Highlight events happening within a specified period (e.g., within the week)	Calendar view	Visual cues for upcoming events	Verify that events are highlighted correctly based on the time filter.																														
5	Implement the ability to edit and delete calendar events		<table><tr><th></th><th>Feature</th><th>Description</th><th>UI Elements</th><th>Database Operation</th><th>Testing</th></tr><tr><td>1</td><td>Edit Event</td><td>Modify the name, date, and time of an existing event.</td><td>Edit button to open an Edit Event modal. Edit button to open an Edit Event modal</td><td>Update event using <code>update_event()</code> method. Data mapping to UI elements</td><td>Ensure the Edit modal populates with current event data. Data mapping to UI elements</td></tr><tr><td>2</td><td>Delete Calendar Event</td><td>Allow users to remove an event from the calendar and database.</td><td>Delete button with confirmation popup</td><td>Delete event using <code>delete_event()</code> method</td><td>Confirm the confirmation popup works and that the event is removed from both the UI and database.</td></tr></table>						Feature	Description	UI Elements	Database Operation	Testing	1	Edit Event	Modify the name, date, and time of an existing event.	Edit button to open an Edit Event modal. Edit button to open an Edit Event modal	Update event using <code>update_event()</code> method. Data mapping to UI elements	Ensure the Edit modal populates with current event data. Data mapping to UI elements	2	Delete Calendar Event	Allow users to remove an event from the calendar and database.	Delete button with confirmation popup	Delete event using <code>delete_event()</code> method	Confirm the confirmation popup works and that the event is removed from both the UI and database.												
				Feature	Description	UI Elements	Database Operation	Testing																													
			1	Edit Event	Modify the name, date, and time of an existing event.	Edit button to open an Edit Event modal. Edit button to open an Edit Event modal	Update event using <code>update_event()</code> method. Data mapping to UI elements	Ensure the Edit modal populates with current event data. Data mapping to UI elements																													
2	Delete Calendar Event	Allow users to remove an event from the calendar and database.	Delete button with confirmation popup	Delete event using <code>delete_event()</code> method	Confirm the confirmation popup works and that the event is removed from both the UI and database.																																
6	Implement the ability to edit, delete, and mark as completed to do list items		<table><tr><th>Feature</th><th>Description</th><th>UI Elements</th><th>Database Operation</th><th>Testing</th></tr><tr><td>Edit Task</td><td>Modify the title, due date, category, and priority of an existing task.</td><td>Edit button to open an Edit Task Modal.</td><td>Update task using <code>update_task()</code> method.</td><td>Ensure modal populates with current task data.</td></tr><tr><td>Delete Task</td><td>Remove a task from the list and the database.</td><td>Delete button with confirmation popup.</td><td>Delete task using <code>delete_task()</code> method.</td><td>Verify confirmation popup works and task is removed.</td></tr><tr><td>Mark Task as Completed</td><td>Toggle the completion status of a task, updating the UI and database.</td><td>Checkbox or toggle to mark task as completed.</td><td>Toggle completion status using <code>update_task()</code> method.</td><td>Check that the completion status is updated correctly in both UI and database.</td></tr></table>					Feature	Description	UI Elements	Database Operation	Testing	Edit Task	Modify the title, due date, category, and priority of an existing task.	Edit button to open an Edit Task Modal.	Update task using <code>update_task()</code> method.	Ensure modal populates with current task data.	Delete Task	Remove a task from the list and the database.	Delete button with confirmation popup.	Delete task using <code>delete_task()</code> method.	Verify confirmation popup works and task is removed.	Mark Task as Completed	Toggle the completion status of a task, updating the UI and database.	Checkbox or toggle to mark task as completed.	Toggle completion status using <code>update_task()</code> method.	Check that the completion status is updated correctly in both UI and database.										
			Feature	Description	UI Elements	Database Operation	Testing																														
			Edit Task	Modify the title, due date, category, and priority of an existing task.	Edit button to open an Edit Task Modal.	Update task using <code>update_task()</code> method.	Ensure modal populates with current task data.																														
			Delete Task	Remove a task from the list and the database.	Delete button with confirmation popup.	Delete task using <code>delete_task()</code> method.	Verify confirmation popup works and task is removed.																														
Mark Task as Completed	Toggle the completion status of a task, updating the UI and database.	Checkbox or toggle to mark task as completed.	Toggle completion status using <code>update_task()</code> method.	Check that the completion status is updated correctly in both UI and database.																																	