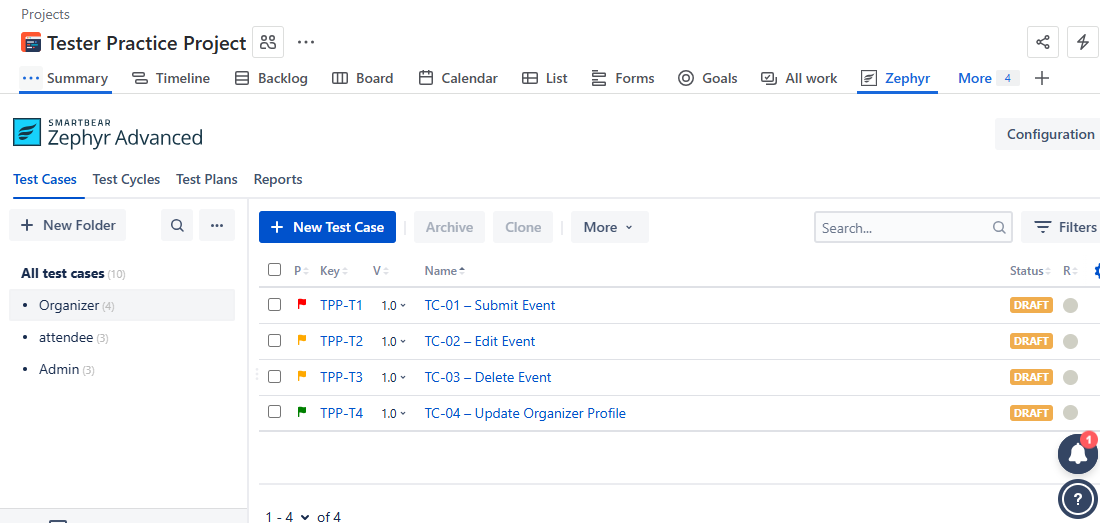
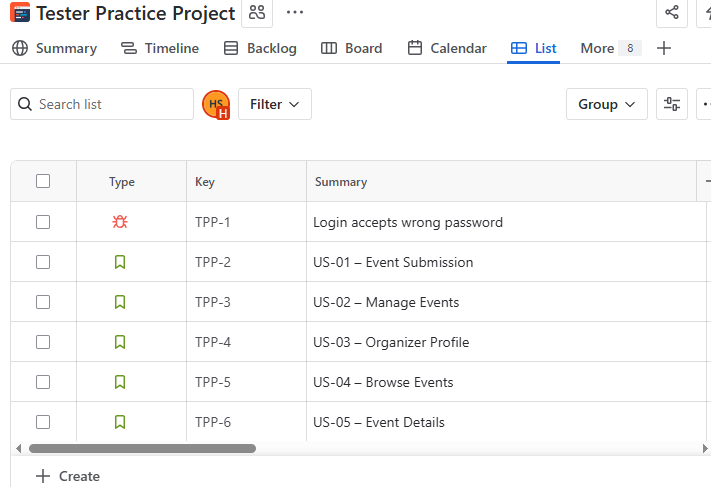
Task 1



Task 2

Defect Lifecycle

When we find a bug in the Darwin Multicultural Events Platform, it will follow these steps:

New / Open – Tester reports the defect in Jira (with steps, screenshot).

Assigned – BA or Team Lead assigns the defect to a developer.

In Progress – Developer starts working on fixing the bug.

Fixed / Resolved – Developer completes the fix and marks the defect as resolved.

Ready for Retest – Tester retests the issue on the updated build.

Closed – If the bug is fixed, tester closes the defect.

Reopened – If the bug still exists, tester reopens it and cycle repeats.

🔹 Severity vs Priority Matrix

We classify bugs using two factors:

Severity (Impact) = how serious the bug is.

Priority (Urgency) = how quickly it must be fixed.

| Severity | Priority | What it Means | Example in Our App |
| --- | --- | --- | --- |
| Critical | P1 – Immediate fix | System is unusable | App crashes when browsing events |
| High | P2 – Fix in sprint | Major feature broken | Cannot submit an event |
| Medium | P3 – Fix soon | Partial issue, workaround exists | Event filter not working correctly |
| Low | P4 – Fix later | Cosmetic/minor | Misaligned button on event page |

Defect Triage (Team Meeting Process)

Defect triage means reviewing all reported bugs with the team and deciding which ones to fix first.

Tester presents defects with severity/priority tags.

BA explains business impact (e.g., “Attendees cannot register, so must fix immediately”).

Developer explains effort and time required.

Team Lead decides final priority and assigns the bug.

Task 3:

CRUD Flow in Our Product

Create – Organizer submits an event.

Check: Event data is saved in the DB correctly.

Read – Attendee browses events.

Check: Event data shown in UI matches what’s in the DB.

Update – Organizer edits an event.

Check: DB updates correctly (old data replaced with new).

Delete – Organizer deletes an event.

Check: Event is removed from DB (or marked inactive).

Example SQL Checks (what QA would run on DB)

1. Create – Insert Event

-- After submitting an event, check if it exists in the DB

SELECT \* FROM Events

WHERE title = 'Diwali Night' AND venue = 'Civic Park';

Expected: 1 row is returned with correct details.

2. Read – Browse Event

-- Check if the event is visible when browsing

SELECT title, date, venue

FROM Events

WHERE isPublished = true;

Expected: The events shown to attendees match the records.

3. Update – Edit Event

-- After editing the venue, check if the change is reflected

SELECT venue

FROM Events

WHERE title = 'Diwali Night';

Expected: Venue shows updated value.

4. Delete – Remove Event

-- After deleting, event should not exist or should be marked inactive

SELECT \*

FROM Events

WHERE title = 'Diwali Night';

Expected: No rows OR row has status = 'inactive'.

Data Integrity Rules

While checking CRUD, always confirm:

No duplicates (two events with same ID shouldn’t exist).

Correct relationships (event must belong to a valid organizer).

Timestamps are accurate (created\_at, updated\_at).

Status flags are correct (e.g., deleted events not shown to attendees).

Task 5

What is UAT?

UAT = testing by real users or stakeholders to confirm the system meets business needs before go-live.

For our project: community members, organizers, or admins will act as testers.

UAT Scripts (step-by-step test cases in plain English)

UAT Script 1 – Submit an Event (Organizer)

Precondition: User is logged in as organizer.

Steps:

Open “Create Event” form.

Enter title, date, venue, cultural tag, and upload an image.

Click Submit.

Expected Result: Event is saved and appears in “My Events” with status “Pending Approval.”

UAT Script 2 – Browse & Filter Events (Attendee)

Precondition: Events exist in system.

Steps:

Go to Browse Events.

Apply filter “Family” + “Weekend.”

Expected Result: Only matching events are displayed.

UAT Script 3 – Approve Event (Admin)

Precondition: An event is pending approval.

Steps:

Open Review Submissions.

Select pending event.

Approve with a note.

Expected Result: Event status changes to “Approved” and becomes visible to attendees.

UAT Data

Prepare realistic dummy data for UAT:

Organizer Test Data:

Event: “Diwali Night” | Date: 2025-11-01 | Venue: Civic Park | Tag: Family, Cultural

Attendee Test Data:

User filters: Family + Weekend | Expected: Diwali Night should appear

Admin Test Data:

Event: “Diwali Night” (Pending) | Action: Approve with note “Reviewed OK”

Train Stakeholders on Defect Reporting

Teach non-technical users how to log bugs clearly:

Where: Jira → Create → Issue type = Bug

Fields to fill:

Summary: Short title (e.g., “Filter not working for Family tag”)

Description: Steps to reproduce + Expected vs Actual

Priority: Select (High, Medium, Low)

Attachments: Screenshot if possible

Sample Defect Report:

Summary: “Browse filter not showing correct results”

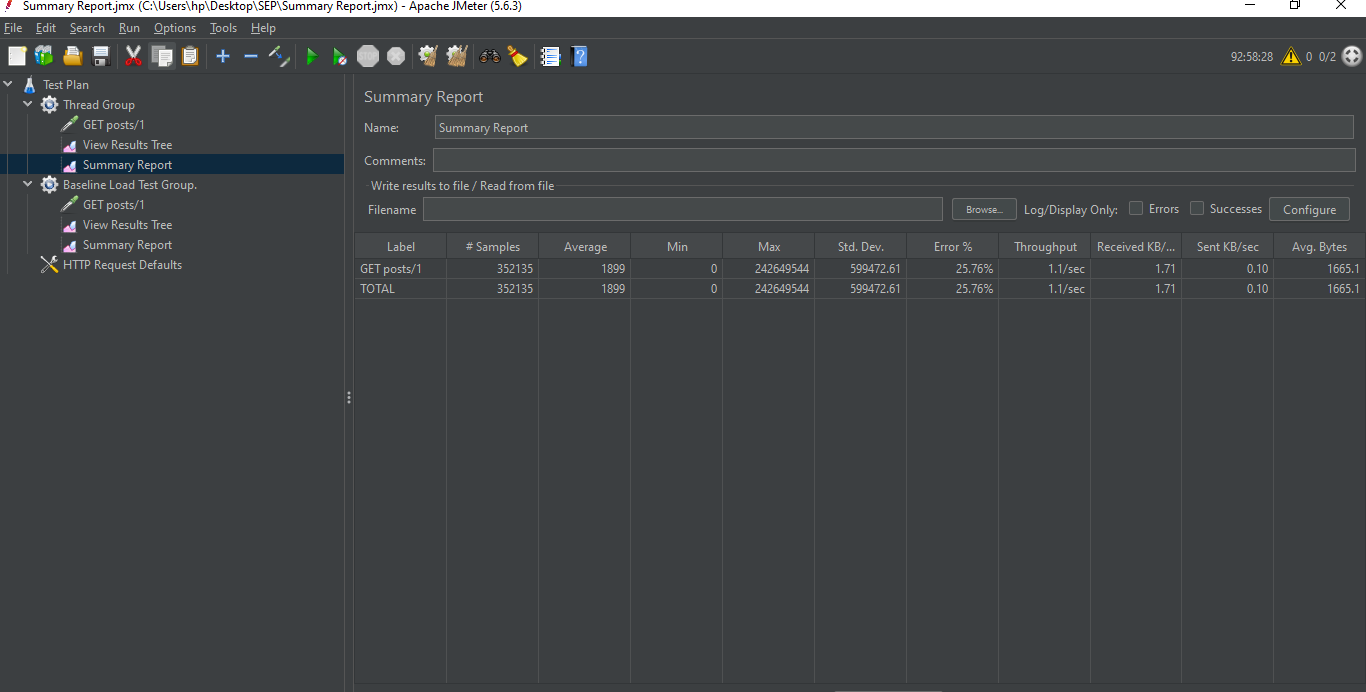
Steps: Go to Browse → Apply Family + Weekend → Still shows all events

Expected: Only family weekend events

Actual: All events shown

Priority: High

Task 6



<https://www.linkedin.com/learning/certificates/587377899bb8cf6a87c03b42aae5684176abb342d6c7b180329fbcd9d628b0ef?trk=share_certificate>

The ISTQB course teaches the basics of software testing in simple steps. It explains why testing is important, different types of testing like manual, automated, functional, and non-functional, and how to write test cases, plan and report bugs, and work with developers. The course follows the official ISTQB syllabus used worldwide, and after completing it, you can take the ISTQB exam to earn a certificate that proves your software testing skills.