Personalized Event Schedule Application

Test Plan Report by: Neubin Sebastian

Test Plan: *Editing and Deleting Events*

1) Test Plan Identifier

TP-EDS-001 – (Test Plan-Edit Delete System-001)

2) Introduction

This test plan outlines the strategy for verifying the functionality of editing and deleting events in the **Personal Event Schedule App**. This feature ensures that only the user who created an event can edit or delete it. The testing will cover **user permissions, UI interactions, data persistence, and error handling**.

3) Test Items

- **Editing an Event**: The user should be able to modify the event title, attendees, and description.

- **Deleting an Event**: The user should be able to delete their own created event.

4) Features to be Tested

- **Edit event** – Users can modify the event **title, attendees, and description**, but not the date and time.
- **Delete event** – Users can remove an event, and it should no longer appear in the calendar.
- **User permissions** – Only the **creator** of the event should be able to edit or delete it.
- **Persistence** – Changes should be saved in the database (db.json).
- **UI interactions** – Buttons should appear only for the **event creator**.

5) Features Not to be Tested

- Event creation (covered separately).
- Login and authentication (handled in a different test).
- Email notifications for event attendees.

6) Approach

The testing will be conducted using:

- **Unit Testing** (Jest & React Testing Library) – To verify UI components and Redux state updates.

- **Automation Testing** (Cypress) – To test event editing and deletion flows in the UI.

- **Manual Testing** – To check UI responsiveness and error handling.

7) Item Pass/Fail Criteria

Pass Criteria

- The **event creator** can edit or delete their event.

- The **edited event details persist** after a page refresh.

- The **deleted event disappears** from the calendar.

- Other users **cannot edit or delete** events they did not create.

Fail Criteria

- A user edits/deletes an event they **did not create**.

- Changes to the event **do not persist** after page refresh.

- Event deletion does **not remove the event from JSON Server**.

8) Suspension Criteria and Resumption Requirements

- Testing will be **paused** if there are major **backend failures** (JSON Server issues).

- Testing will resume once the backend is fixed or a mock service is implemented.

9) Test Deliverables

- **Test cases document** (Jest & Cypress test scripts).
- **Test execution report** (pass/fail status for each test case).
- **Bug report** (if any issues are found).

10) Testing Tasks

- Write **unit tests** for editing and deleting events using Jest.

- Implement **Cypress tests** for automated UI validation.

- Perform **manual testing** to verify UI consistency.

- Log and report **bugs** if any unexpected behavior is found.

11) Environmental Needs

- **Browser**: Google Chrome (latest version).

- **Tools**: Jest, React Testing Library, Cypress.

- **Test Data**: User accounts, pre-created events in db.json.

12) Responsibilities

- **Frontend Developer**: Ensures UI and Redux state updates correctly.

- **QA Tester**: Runs Jest, Cypress tests and reports bugs.

- **Team Lead**: Reviews test results and approves fixes.

13) Staffing and Training Needs

- Familiarity with Jest & Cypress for writing tests.

- Training session on Redux state testing if required.

14) Schedule

Task	Duration	Assigned To
Write unit tests	1 day	Developer
Implement Cypress tests	2 days	QA Tester
Manual Testing	1 day	Tester
Bug Fixes & Review	1 day	Team Lead
Final Approval	1 day	Team Lead

15) Risk and Contingencies

Risk	Impact	Mitigation
JSON Server crashes	High	Use a mock API for testing.
UI does not update after editing	Medium	Ensure Redux state updates correctly.
Unauthorized users can delete events	High	Fix permissions in code.

16) Approvals

- Project Lead: *Neubin Sebastian*

- QA Tester: *Neubin Sebastian*

- Professor: *Prof. Peter Dillinger*