Test Plan & Strategy Document

For React Frontend & Node.js Backend CRUD Application

1. Introduction

This document outlines the testing approach for a small web application featuring a React.js frontend and a Node.js Express backend API. The app supports user authentication and basic CRUD operations for items (e.g., todo list), with a focus on both user interfaces and API endpoints.

2. What is Being Tested

- User Interface (UI):
  - Authentication flow (Login/Logout)
  - Item operations: Create, Read, Update, Delete (CRUD)
  - UI state updates: Successful operation feedback, error messages, and data displays
- API Layer:
  - Authentication endpoint
  - Items CRUD endpoints
  - Input validation and appropriate error handling

3. Test Coverage Areas

Area	Cases Included
Authentication	Valid/invalid login, UI feedback, protected route access
CRUD Operations	Create, retrieve, update, delete item; reflected UI changes
Error Handling	Invalid input, missing fields, unauthorized access, not found
API Status Codes	200/201 on success, 400/401/404 on failures
Response Content	JSON structure, data integrity after actions
UI State	List rendering, form resets, notifications/messages

4. Tools Used and Rationale

Tool	Usage	Reason for Selection
Cypress	UI end-to-end (E2E) automation	Fast, reliable, supports real browser automation, friendly syntax for React apps
Jest	Backend API/unit/integration tests	Popular in Node.js ecosystem, feature-rich, supports mocks & coverage

Supertest	HTTP assertions with Jest	Makes HTTP calls/test validations simple for Express APIs

5. How to Run the Tests

Backend API Tests

1. In backend folder, install dependencies:
   - npm install

2. Run backend tests:
   - npm test

Frontend UI Tests

1. In frontend folder, install dependencies:
   - npm install

2. Start React app:
   - npm start

3. In another terminal, open Cypress:
   - npx cypress open

6. Assumptions & Limitations

- Hardcoded user for authentication.
- In-memory data storage (not persistent).
- Both apps run locally (localhost:3000 and localhost:5000).
- Accessibility, performance, and security not in scope.
- Tests require manual data reset on each run as needed.