# Test Plan – Todo App (React + Node.js)

### Project Overview

This test plan outlines the testing approach for a basic web application built using:

React (frontend) for the UI

Node.js Express (backend) with in-memory data (no database)

The purpose is to demonstrate functional UI and REST API test automation using Selenium and REST-assured.

### What Is Being Tested

Functional Scope:

Authentication via /login endpoint

CRUD operations on "todo items":

View (GET /items)

Create (POST /items)

Update (PUT /items/:id)

Delete (DELETE /items/:id)

UI Scope:

Login flow

Adding a todo item

Editing a todo item

Deleting a todo item

Verifying item visibility and updates in the UI

### Tools & Technologies Used

| Tool | Purpose |
| --- | --- |
| Selenium | UI automation (React frontend) |
| REST-assured | API testing (Node.js backend) |
| TestNG | Test execution framework |
| Maven | Dependency and test lifecycle manager |
| ChromeDriver | Browser automation for Selenium |

### Test Coverage Areas

UI Automation with Selenium

|  |
| --- |
| Login with valid credentials |
| Login with invalid credentials |
| Create new todo item |
| Edit existing todo item |
| Delete todo item |
| Assert presence/absence of item |

### API Automation with REST-assured

| API Endpoint | Test Cases Covered |
| --- | --- |
| POST /login | Valid & Invalid Login |
| GET /items | Fetch items |
| POST /items | Add item (valid) |
| PUT /items/:id | Update item (valid/invalid) |
| DELETE /items/:id | Delete item (valid/invalid) |

### How to Run the Tests

UI Tests (Selenium)

Start backend (node index.js)

Start frontend (npm start)

Run tests:

mvn test -Dtest=TodoUITest

API Tests (REST-assured)

mvn test -Dtest=TodoApiTest

Make sure backend is running at http://localhost:5000 before executing.

### Assumptions and Limitations

Backend does not persist data (in-memory only); test data is lost between runs.

No token-based auth used; login is stateless and does not return a token.

Tests assume the app is running locally at:

http://localhost:3000 (frontend)

http://localhost:5000 (backend)

Selenium uses ChromeDriver — Chrome must be installed.

Only basic validation is implemented on backend (e.g., empty item text not rejected).