 

**COLLEGE CODE : 9623**

**COLLEGE NAME:** Amrita College Of Engineering And

Technology, Erachakulam

**DEPARTMENT :** COMPUTER SCIENCE ENGINEERING

**STUDENT NM- ID :** autcs0085

**ROLL NO :** 962323104085

**DATE :**24/09/2025

**C**ompleted **T**he **P**roject **N**amed **A**s **P**hase**\_ T**echnology

**PROJECT NAME : SINGLE PAGE APPLICATION**

**S**ubmitted **B**y**,**

**NAME:** RATHEESH.G.K(962323104085)

MOBILE NO:+91 9488179808

**MVP Implementation Plan – Single Page Application:**

**1. Project Setup**

* **Tech stack selection**:
  + Frontend: React (SPA framework)
  + State management: React Context / Redux (if complex state)
  + Styling: TailwindCSS / Material UI
  + Backend (if required): Node.js + Express
  + Database (if required): MongoDB / PostgreSQL (for persistent storage).
* **Setup steps**:

npx create-react-app spa-mvp # or: npm create vite@latest spa-mvp -- template react

cd spa-mvp

npm install axios react-router-dom

npm install -D jest @testing-library/react

git init

**2. Core Features Implementation**

* Identify MVP features (example: User Authentication, Dashboard, CRUD for items).
* Routing with react-router-dom:
  + /login → Login Page
  + /dashboard → Dashboard Page
  + /items → CRUD Operations
* **Components**:
  + Navbar, Footer, Form, Card, Modal
* **State management**:
  + Local state with React hooks
  + Context API (for global state, e.g., user auth session)

**3. Data Storage:**

* Store data in React state / Context.
* Use local Storage for persistence:

**Option A – Local State (for early MVPs, no backend)**

useEffect(() => {

local Storage.setItem("items", JSON stringify (items));

}, [items]);

**Option B – Database (for scalable MVPs):**

**Setup backend API with Express:**

app.get('/api/items', (req, res) => { ... });

app.post('/api/items', (req, res) => { ... });

* Connect to MongoDB/PostgreSQL for persistence.
* Consume APIs with axios in React.

**4. Testing Core Features:**

test("renders login form", () => {

render(<Login />);

expect(screen.getByLabelText(/username/i)).toBeInTheDocument();

});

* **Integration Tests**: test API calls with msw (Mock Service Worker).
* **E2E Tests** (optional at MVP stage) with Cypress/Playwright.

**5. Version Control (GitHub):**

git init

git add .

git commit -m "Initial commit: project setup"

git branch -M main

git remote add origin https://github.com/username/spa-mvp.git

git push -u origin main

**Workflow:**

* **Feature branches** → feature/login-page
* **Pull requests** for merging
* Use **GitHub Projects/Issues** for task tracking