Milestone 2 – Assessment Set 1

Title: *React Product Dashboard with Express.js Backend Integration*

Duration: 120 Minutes | **Total Marks:** 60
Mode: Individual | **Submission Format:** ZIP folder / PDF Report

---

Objective

To evaluate participants' ability to build an interactive React front-end application integrated with a mock backend (json-server / Express.js) demonstrating routing, API handling, form validation, and component-based design.

Assessment Overview

User Story	**Focus Area**	**Complexity**	**Marks**
User Story 1	React Components, Props, State, Event Handling	Easy	20
User Story 2	React Router, API Integration (GET, POST)	Medium	20
User Story 3	Formik + Yup Form Handling, Context API	Difficult	20
Total			**60 Marks**

User Story 1 – Product Catalog (React Basics)

Scenario:
Build a simple React component-based Product Catalog that lists products with name, price, and category. **Requirements:**

- Use create-react-app to scaffold the project.

- Create **ProductCard** (functional) and **ProductList** (class) components.
- Use **props** for data passing and **state** for managing favorite status.
- Apply **Bootstrap** for styling (cards, grids).

User Story 2 – Product Details with Routing

Scenario:
Integrate **React Router** to navigate between Product List and Product Detail pages

using a backend API. **Requirements:**

- Use json-server or Express.js backend with product data in JSON format.
- Implement GET /products and GET /products/:id.
- Handle loading and error states using try...catch.
- Demonstrate **Route Transition** between pages.

User Story 3 – Add Product Form using Formik + Yup

Scenario:
Create a form to add a new product to the database. **Requirements:**

- Use **Formik** and **Yup** for controlled form and validation.
- Fields: name, price, category, description.
- On submit → POST data to API.
- Use **Context API** for global product list update.
- Add basic form styling with Bootstrap.

Bonus Challenge (Optional – 5 Marks Extra)

Implement **Lazy Loading** for Product Detail Page using React.lazy() and Suspense.

Submission Format

- /src/components/ → React components
- /src/context/ → Context provider file
- /backend/server.js → Express server (if used)
- Screenshots of:
    - Product list
    - Product detail page
    - Add form validation
