

Milestone 2 – Product Dashboard Report

Student Name: *Amirtha Varshini S*

Role: Full Stack MERN Developer Trainee

Assessment: Milestone 2 – React Product Dashboard with Express Backend

Date: 25/11/2025

Introduction

This milestone focuses on building a React-based Product Dashboard with routing, backend mock API integration, product detail pages, context-based global state sharing, and form handling using Formik with Yup validation. The project uses a manual folder structure with a backend server built using Express.js and an interactive UI built using React and Bootstrap.

The goal of this project is to demonstrate core Frontend and Backend development skills such as:

- Component architecture (class & functional components)
 - Routing & navigation
 - REST API integration
 - Form validation & submission
 - State management using Context API
 - Error handling & loading states
 - Lazy loading (Bonus marks)
-

Folder Structure

```
milestone-2-product-dashboard/
  frontend/
    backend/
      server.js
  src/
    components/
    context/
    App.js
    index.js
  screenshots/
  README.md
```

Technologies Used

Frontend	Backend	Other Tools
React.js	Node.js & Express.js	Axios
React Router	In-memory dataset API	Bootstrap
Formik & Yup	CORS	Google Docs
Context API	REST handlers	VS Code, Postman

User Story Implementations

User Story 1 – Display Product Catalog with Components

Requirement

- Display list of products using React components & Bootstrap styling
- Use **class and functional components**, props, state & event handling
- Add **favorite/unfavorite** functionality

Implementation Summary

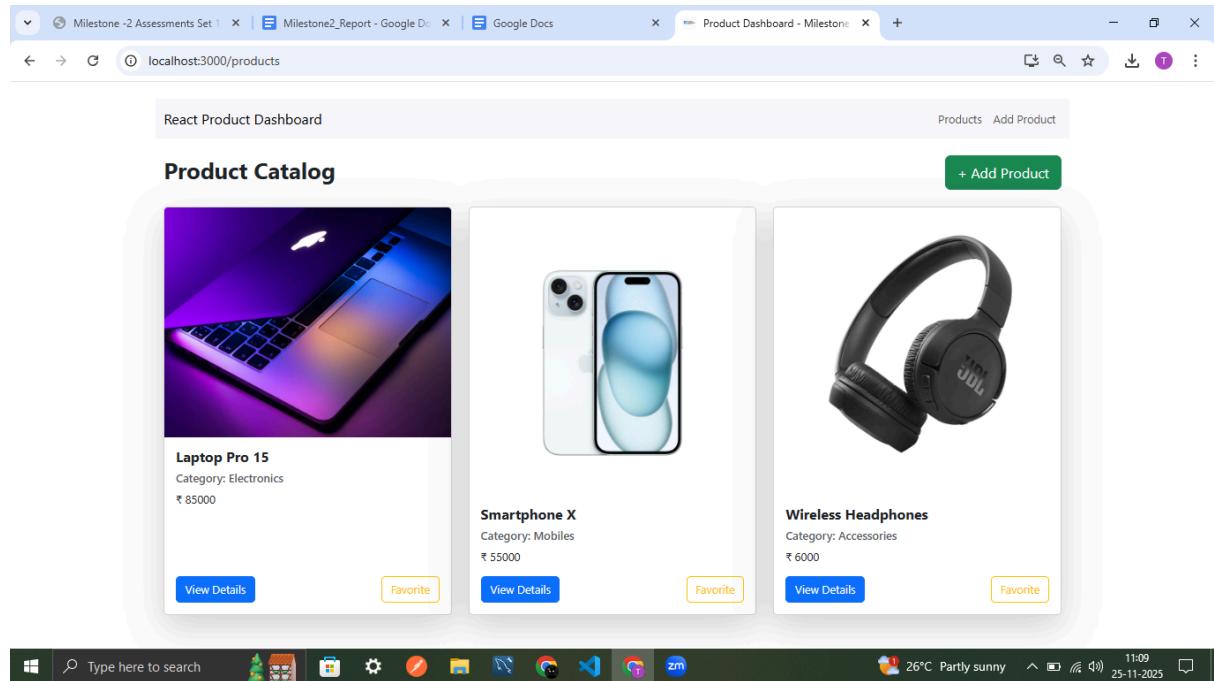
- `ProductList.jsx` created as a **Class Component**
- `ProductCard.jsx` created as a **Functional Component**
- Favorite toggle handled using state
- Bootstrap grid UI design

Relevant Files

- `src/components/ProductList.jsx`
- `src/components/ProductCard.jsx`

Output Screenshot

(*Q1-product-list.png*)



User Story 2 – Product Detail Page with React Router

Requirement

- Dynamic product detail page using URL parameters
- Load details using backend API with loading & error handling
- Implement API call using Axios

Implementation Summary

- Used React Router: `Route path="/products/:id"`
- Detail page built with Handling loading / error UI
- API call: `GET /products/:id`

Relevant Files

- `App.js`
- `ProductDetail.jsx`
- `backend/server.js`

Output Screenshot

(Q2-product-detail.png)

The screenshot shows a browser window with multiple tabs open. The active tab is titled 'Product Dashboard - Milestone2_Report' and has the URL 'localhost:4000/products'. The page content is a 'React Product Dashboard' showing a 'Product Detail' for a product named 'Wireless Headphones'. The product details include: Category: Accessories, Price: ₹ 6000, and Description: Noise cancelling over-ear headphones. A 'Back to Products' button is visible at the bottom left of the detail card. The top right of the dashboard header has links for 'Products' and 'Add Product'. The browser's address bar shows the URL 'localhost:3000/products/3'.



User Story 3 – Form for Adding New Product

Requirement

- Add product form built using Formik + Yup for validation
- Submit using **POST /products** API
- Update global state through **Context API**

Implementation Summary

- Created `AddProductForm.jsx`
- Form validation using Yup schemas
- On success, product is added & list refreshed

Relevant Files

- `src/components/AddProductForm.jsx`
- `src/context/ProductContext.jsx`
- `backend/server.js`

Output Screenshots

(Q3-add-form-success.png)

A screenshot of a web browser showing a successful product addition. The browser has multiple tabs open, including 'Milestone -2 Assessm...', 'Milestone2_Report...', 'Google Docs', 'Product Dashboard', 'Amazon Basics 3-in-1...', and 'Amazon.in : gadgets...'. The main window displays a 'React Product Dashboard' with a 'Products' and 'Add Product' button. A modal window titled 'Add New Product' is open, containing fields for Product Name ('Three-in-one USB C HUB multiport'), Price ('569'), Category ('Computers & Accessories'), Description ('USB-C PD-IN fast charging port?The USB C adapter support up to 100 W power transmission to your Laptop. Can only charge your laptop through Power delivery. Cannot take reverse charging to charge your Phone/power bank or any other device through Type-C port.'), and Image URL ('https://m.media-amazon.com/images/I/51t09bjNM3L_SX679.jpg'). A green 'Add Product' button is visible on the right.

(Q3-add-form-validation.png)

A screenshot of a web browser showing validation errors for a product addition form. The browser tabs and dashboard are identical to the previous screenshot. The 'Add New Product' modal shows validation messages: 'Price must be a number' for the Price field (containing '\$ 569') and 'Category is required' for the Category field (empty). The other fields (Product Name, Description, and Image URL) are valid. The green 'Add Product' button is present.

Bonus Feature – Lazy Loading

Requirement

- Implement React.lazy & Suspense

Summary

- Product Detail page loaded dynamically to improve performance

Relevant Code Snippet

```
const ProductDetail = lazy(() =>
import("./components/ProductDetail"));
```

Backend API Endpoints

Method	Endpoint	Description
GET	/products	Returns all products
GET	/products/:id	Returns single product
POST	/products	Adds new product

Backend Running Screenshot

File Edit Selection View Go Run ... < > milestone-2-product-dashboard

frontend backend server.js products

```
12 let products = [
13   {
14     name: "Samsung Galaxy S21",
15     price: 55000,
16     category: "Mobiles",
17     description: "Flagship smartphone with AMOLED display.",
18     image: "https://m.media-amazon.com/images/I/71d7rf5l0wl..SL1500_.jpg"
19   },
20   {
21     id: 3,
22     name: "Wireless Headphones",
23     price: 6000,
24     category: "Accessories",
25     description: "Noise cancelling over-ear headphones.",
26     image: "https://m.media-amazon.com/images/I/61kFL7ywS2..SL1500_.jpg"
27   }
28 ]
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\AMIRTHAVARSHINI\Documents\GitHub\Wipro-NGA-Training\milestone-2-product-dashboard\frontend\backend>
PS C:\Users\AMIRTHAVARSHINI\Documents\GitHub\Wipro-NGA-Training\milestone-2-product-dashboard\frontend\backend> node server.js
My Backend server running on http://localhost:4000
PS C:\Users\AMIRTHAVARSHINI\Documents\GitHub\Wipro-NGA-Training\milestone-2-product-dashboard\frontend\backend> node server.js
My Backend server running on http://localhost:4000
PS C:\Users\AMIRTHAVARSHINI\Documents\GitHub\Wipro-NGA-Training\milestone-2-product-dashboard\frontend\backend> node server.js
My Backend server running on http://localhost:4000
```

main* Connect Amirtha_wipro_D8 edupro

Ln 31, Col 33 Spaces: 4 UTF-8 CRLF { } JavaScript Go Live

Type here to search

Testing & Verification

- Manual testing performed in browser and console
 - Verified routing navigation, error states & form validation
 - Verified data persistence in state after POST request

Screenshots Folder

1. Product Catalog – `03-product-catalog.png`
 2. Product Detail – `04-product-detail.png`
 3. Add Product Form – `05-add-product-form-empty.png`
 4. Validation Errors – `06-add-product-validation-errors.png`
 5. After Adding Product – `07-product-list-after-add.png`
 6. Backend Server – `01-backend-server-running.png`
-

Conclusion

This milestone project successfully demonstrates:

- ✓ Understanding of React component architecture
- ✓ Logical routing structure and UI navigation
- ✓ REST API communication using Axios
- ✓ Form implementation with validation
- ✓ Global state sharing using Context API
- ✓ Backend handling via Express.js
- ✓ Lazy loading optimization

I gained practical hands-on experience building production-level MERN application structure and strengthened essential skills required for professional full-stack development.

Submitted By

Name: Amirtha Varshini S

Milestone: 2 – Completed Successfully