Milestone 2

Frontend - npx create-react-app frontend

Install dependencies – npm install axios formik yup react-router-dom bootstrap

For run – npm start

Backend – npm init -y

Install dependencies – npm install express cors

Run --node server .js

Use axios to fetch api data

Add lazyloading

User Story 1 — React Basics

- ProductCard.js
- ProductList.js
- app.css

User Story 2 — Routing + API

- server.js
- App.js
- ProductDetail.js
- ProductContext.js

User Story 3 — Formik + Yup + POST

- AddProductForm.js
- ProductContext.js
- server.js

Bonus — Lazy Loading

- App.js

Backend ----

```
// backend/server.js
// Product Dashboard
// Provides GET (all, single) + POST (add product)
// In-memory database for demo

const express = require("express");
const cors = require("cors");

const app = express();
const PORT = 5000;

// Middlewares
app.use(cors());
app.use(express.json());

// Mock DB for demo
let products = [
 {
   id: 1,
   name: "Wireless Headphones",
   price: 1999,
   category: "Electronics",
   description: "Over-ear headphones with noise cancellation."
```

```
  },
  {
    id: 2,
    name: "Running Shoes",
    price: 2499,
    category: "Footwear",
    description: "Comfortable daily running shoes."
  }
];

// Default route
app.get("/", (req, res) => {
  res.send("Product API running...");
});

// GET all products
app.get("/products", (req, res) => {
  res.json(products);
});

// GET product by ID
app.get("/products/:id", (req, res) => {
  const id = Number(req.params.id);
  const product = products.find((p) => p.id === id);

  if (!product) {
```

```
    return res.status(404).json({ error: "Product not found" });
  }

  res.json(product);
});

// POST add new product
app.post("/products", (req, res) => {
  const { name, price, category, description } = req.body;

  if (!name || !price || !category || !description) {
    return res.status(400).json({ error: "All fields required" });
  }
// Create new product
  const newProduct = {
    id: products.length + 1,
    name,
    price,
    category,
    description
  };

  products.push(newProduct);
  res.status(201).json(newProduct);
});
```

```
// Start server
app.listen(PORT, () => {
  console.log(`Backend running at http://localhost:${PORT}`);
});
```

Frontend –

Components –

Addproducts Form –

```
// Form to add product using Formik + Yup validation
// Updates product list using Context API

import React, { useContext } from "react";
import { useFormik } from "formik";
import * as Yup from "yup";
import { ProductContext } from "../context/ProductContext";

const AddProductForm = () => {
  const { addProduct } = useContext(ProductContext);

  const formik = useFormik({
    initialValues: {
      name: "",
```

```
    price: "",
    category: "",
    description: ""
  },
  validationSchema: Yup.object({
    name: Yup.string().required("Required"),
    price: Yup.number().required("Required"),
    category: Yup.string().required("Required"),
    description: Yup.string().required("Required")
  }),
  onSubmit: (values, { resetForm }) => {
    addProduct(values);
    resetForm();
    alert("Product added!");
  }
});

return (
  <div className="container mt-4">
    <h2>Add Product</h2>

    <form onSubmit={formik.handleSubmit} className="mt-3">

      <input
        type="text"
        name="name"
```

```
  className="form-control mb-2"
  placeholder="Product Name"
  {...formik.getFieldProps("name")}
/>
<small className="text-danger">{formik.errors.name}</small>

<input
  type="number"
  name="price"
  className="form-control mb-2"
  placeholder="Price"
  {...formik.getFieldProps("price")}
/>
<small className="text-danger">{formik.errors.price}</small>

<input
  type="text"
  name="category"
  className="form-control mb-2"
  placeholder="Category"
  {...formik.getFieldProps("category")}
/>
<small className="text-danger">{formik.errors.category}</small>

<textarea
  name="description"
```

```
          className="form-control mb-2"
          placeholder="Description"
          rows="3"
          {...formik.getFieldProps("description")}
        />
        <small className="text-danger">{formik.errors.description}</small>

        <button type="submit" className="btn btn-success mt-2">
          Add Product
        </button>
      </form>
    </div>
  );
};

export default AddProductForm;
```

productCard –

```
// Shows a product in card format
// Handles favorite state using props

import React from "react";
import { Link } from "react-router-dom";
// ProductCard component
```

```
const ProductCard = ({ product, isFav, toggleFav }) => {
  return (
    <div className="col-md-4 mb-4">
      <div className="card p-3 shadow-sm">
        <h5>{product.name}</h5>
        <small className="text-muted">{product.category}</small>

        <p className="mt-2 text-truncate">{product.description}</p>

        <div className="d-flex justify-content-between">
          <strong>₹ {product.price}</strong>

          <button
            className={`btn btn-sm ${
              isFav ? "btn-warning" : "btn-outline-secondary"
            }`}
            onClick={() => toggleFav(product.id)}
          >
            {isFav ? "★" : "☆"}
          </button>
        </div>

        <Link to={`/products/${product.id}`} className="btn btn-primary btn-sm
mt-3">
          View Details
        </Link>
```

```
      </div>
    </div>
  );
};

export default ProductCard;
```

product details –

```
// Display single product detail fetched from backend

import React, { useEffect, useState } from "react";
import { useParams, Link } from "react-router-dom";
import axios from "axios";
// ProductDetail component
const ProductDetail = () => {
  const { id } = useParams();
  const [product, setProduct] = useState(null);
// Fetch product
  useEffect(() => {
    axios
      .get(`http://localhost:5000/products/${id}`)
      .then((res) => setProduct(res.data))
      .catch(() => setProduct(null));
  }, [id]);
// Render product detail or error message
```

```
  if (!product) return <p className="text-center mt-4">Product not
found...</p>;


  return (
    <div className="container mt-4">
      <div className="card p-4 shadow-sm">
        <h2>{product.name}</h2>
        <p className="text-muted">{product.category}</p>
        <p>{product.description}</p>
        <h4>₹ {product.price}</h4>


        <Link className="btn btn-secondary mt-3" to="/">
          Back
        </Link>
      </div>
    </div>
  );
};

export default ProductDetail;
```

productlist—

```
// Class component: Manages local favorite state
// Gets product list from Context API
```

```
import React, { Component } from "react";
import { ProductContext } from "../context/ProductContext";
import ProductCard from "./ProductCard";
// ProductList component to display list of products
class ProductList extends Component {
  static contextType = ProductContext;

  state = {
    favorites: {}
  };
// Toggle favorite state for each product
  toggleFav = (id) => {
    this.setState({
      favorites: {
        ...this.state.favorites,
        [id]: !this.state.favorites[id]
      }
    });
  };

  render() {
    const { products } = this.context;
// Render products
    return (
      <div className="container mt-4">
        <h2>Product Catalog</h2>
```

```
      <div className="row mt-3">
        {products.map((p) => (
          <ProductCard
            key={p.id}
            product={p}
            isFav={this.state.favorites[p.id]}
            toggleFav={this.toggleFav}
          />
        ))}
      </div>
    </div>
  );
 }
}

export default ProductList;
```

context –

productcontext.js-

```
// Context to store products globally and refresh after adding a new one

import React, { createContext, useState, useEffect } from "react";
import axios from "axios";

export const ProductContext = createContext();
```

```
// Provider
export const ProductProvider = ({ children }) => {
  const [products, setProducts] = useState([]);

  // Load products once
  useEffect(() => {
    axios.get("http://localhost:5000/products").then((res) => {
      setProducts(res.data);
    });
  }, []);
// Add product
  const addProduct = async (product) => {
    const res = await axios.post("http://localhost:5000/products", product);
    setProducts([...products, res.data]);
  };
// Return provider
  return (
    <ProductContext.Provider value={{ products, addProduct }}>
      {children}
    </ProductContext.Provider>
  );
};
```

App.js

```
// Handles all routes and lazy loads the ProductDetail page
```

```
import React, { Suspense, lazy } from "react";
import { Routes, Route, Link } from "react-router-dom";
import ProductList from "./components/ProductList";
import AddProductForm from "./components/AddProductForm";
// Lazy load ProductDetail
const ProductDetail = lazy(() => import("./components/ProductDetail"));

function App() {
  // Render app layout
  return (
    <>
      {/* Simple Navigation Bar */}
      <nav className="navbar navbar-dark bg-dark px-3">
        <Link className="navbar-brand" to="/">
          Product Dashboard
        </Link>
        <div>
          <Link to="/" className="btn btn-outline-light btn-sm mx-2">
            Catalog
          </Link>
          <Link to="/add" className="btn btn-success btn-sm">
            Add Product
          </Link>
        </div>
      </nav>
```

```
      {/* Routes */}
      <Suspense fallback={<p className="text-center mt-4">Loading page...</p>}>
        <Routes>
          <Route path="/" element={<ProductList />} />
          <Route path="/add" element={<AddProductForm />} />
          <Route path="/products/:id" element={<ProductDetail />} />
        </Routes>
      </Suspense>
    </>
  );
}

export default App;
```

index .js---

```
//  wraps App in Router and Context Provider

import React from "react";
import ReactDOM from "react-dom/client";
import App from "./App";
```

```
import { BrowserRouter } from "react-router-dom";
import { ProductProvider } from "./context/ProductContext";
import "bootstrap/dist/css/bootstrap.min.css";
import "./styles/app.css";

const root = ReactDOM.createRoot(document.getElementById("root"));

root.render(
  <BrowserRouter>
    <ProductProvider>
      <App />
    </ProductProvider>
  </BrowserRouter>
);
```