Product Dashboard - Implementation Documentation Report

Project Overview

Project: React Product Dashboard with Express.js Backend Integration
Technology Stack: React, Express.js, Formik, Yup, Bootstrap, Context API

User Story 1 - Product Catalog (React Basics)

Implementation Approach

1. **Project Scaffolding**: Used create-react-app for initial setup
2. **Component Architecture**: Implemented both functional and class components
3. **State Management**: Used React state for favorite functionality
4. **Data Flow**: Props for parent-child communication
5. **Styling**: Bootstrap for responsive grid and card layouts

Key Code Snippets

ProductCard.js (Functional Component)

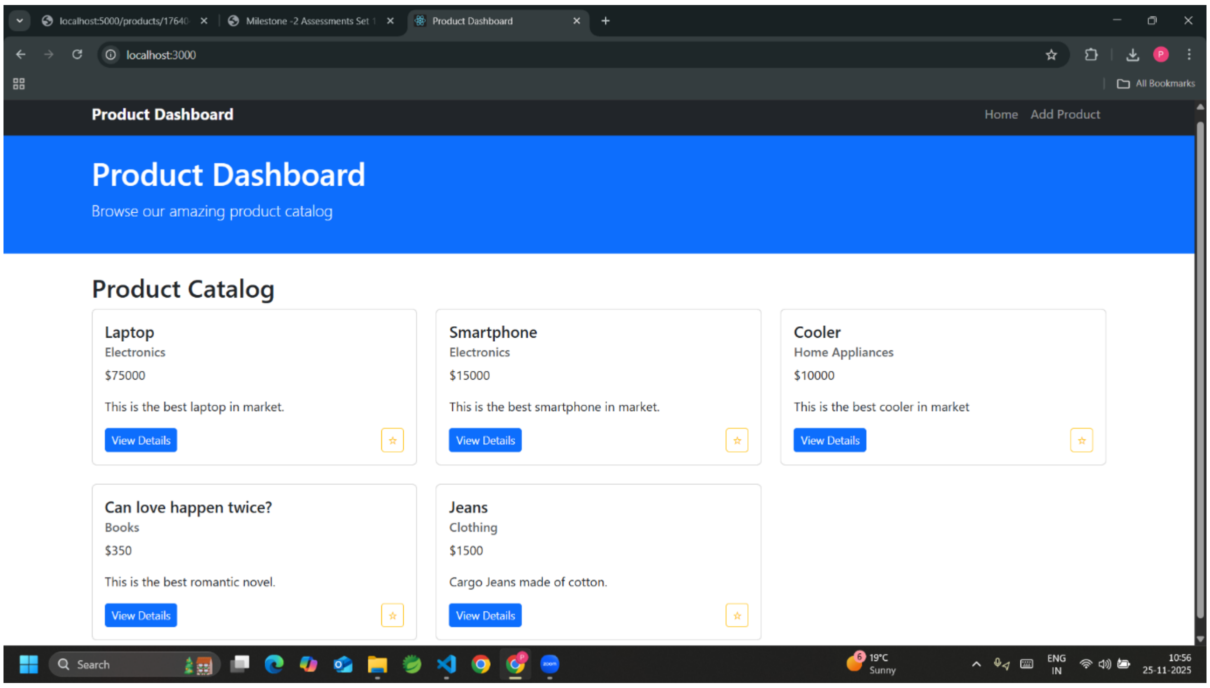
```
import React, { useState } from 'react';
import { Link } from 'react-router-dom';

const ProductCard = ({ product }) => {
    const [isFavorite, setIsFavorite] = useState(false);

    const toggleFavorite = () => {
        setIsFavorite(!isFavorite);
    };

    return (
        <div className="col-md-4 mb-4">
            <div className="card h-100">
                <div className="card-body">
                    <h5 className="card-title">{product.name}</h5>
                    <h6 className="card-subtitle mb-2 text-
muted">{product.category}</h6>
                    <p className="card-text">${product.price}</p>
                    <p className="card-text text-
truncate">{product.description}</p>

                    <div className="d-flex justify-content-between align-
items-center">
                        <Link to={`/product/${product.id}`} className="btn
btn-primary btn-sm">
```

```
                            View Details
                        </Link>
                        <button
                            className={`btn btn-sm ${isFavorite ? 'btn-
warning' : 'btn-outline-warning'}`}
                            onClick={toggleFavorite}
                        >
                            {isFavorite ? '★' : '☆'}
                        </button>
                    </div>
                </div>
            </div>
        </div>
    );
};

export default ProductCard;
```

ProductList.js (Class Component)

```
import React, { Component } from 'react';
import ProductCard from './ProductCard';
import axios from 'axios';

class ProductList extends Component {
    constructor(props) {
        super(props);
        this.state = {
            products: [],
            loading: true,
            error: null
        };
    }

    async componentDidMount() {
        try {
            const response = await
axios.get('http://localhost:5000/products');
            this.setState({
                products: response.data,
                loading: false
            });
        } catch (error) {
            this.setState({
                error: 'Failed to load products',
                loading: false
            });
        }
    }

    render() {
        const { products, loading, error } = this.state;

        if (loading) {
            return (
                <div className="d-flex justify-content-center">
                    <div className="spinner-border" role="status">
                        <span className="visually-hidden">Loading...</span>
                    </div>
                </div>
            );
        }

        if (error) {
            return (
                <div className="alert alert-danger" role="alert">
                    {error}
                </div>
            );
        }

        return (
            <div className="container mt-4">
                <h2>Product Catalog</h2>
                <div className="row">
                    {products.map(product => (
                        <ProductCard key={product.id} product={product} />
                    ))}
                </div>
            </div>
        );
    }
}

export default ProductList;
```

Output


localhost:3000

Product Dashboard — Home — Add Product

Product Dashboard

Browse our amazing product catalog

Product Catalog

Laptop
Electronics
$75000
This is the best laptop in market.
View Details

Smartphone
Electronics
$15000
This is the best smartphone in market.
View Details

Cooler
Home Appliances
$10000
This is the best cooler in market
View Details

Can love happen twice?
Books
$350
This is the best romantic novel.
View Details

Jeans
Clothing
$1500
Cargo Jeans made of cotton.
View Details

Technical Achievements

- Functional component with useState hook
- Class component with lifecycle methods
- Props drilling between components
- Event handling for favorite toggle
- Bootstrap grid system implementation
- Loading and error state management

User Story 2 - Product Details with Routing

Implementation Approach

1. **Routing Setup:** React Router DOM for navigation
2. **API Integration:** Axios for HTTP requests
3. **Dynamic Routing:** Route parameters for product IDs
4. **Error Handling:** Try-catch blocks with user feedback
5. **Loading States:** Visual indicators during API calls

Key Code Snippets

Backend API (Express.js)

```
const express = require('express');
const cors = require('cors');
const fs = require('fs');
```

```javascript
const path = require('path');

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

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

const productsFile = path.join(__dirname, 'products.json');

// Helper function to read products
const readProducts = () => {
    try {
        const data = fs.readFileSync(productsFile, 'utf8');
        return JSON.parse(data);
    } catch (error) {
        return { products: [] };
    }
};

// Helper function to write products
const writeProducts = (products) => {
    fs.writeFileSync(productsFile, JSON.stringify(products, null, 2));
};

// Initialize sample data if file doesn't exist
if (!fs.existsSync(productsFile)) {
    const initialProducts = {
        products: [
            {
                id: 1,
                name: "Laptop",
                price: 999.99,
                category: "Electronics",
                description: "High-performance laptop"
            },
            {
                id: 2,
                name: "Smartphone",
                price: 699.99,
                category: "Electronics",
                description: "Latest smartphone model"
            },
            {
                id: 3,
                name: "Coffee Maker",
                price: 89.99,
                category: "Home Appliances",
                description: "Automatic coffee maker"
            }
        ]
    };
    writeProducts(initialProducts);
}

// GET /products - Get all products
app.get('/products', (req, res) => {
    try {
        const data = readProducts();
        res.json(data.products);
    } catch (error) {
        res.status(500).json({ error: 'Failed to fetch products' });
    }
});

// GET /products/:id - Get single product
app.get('/products/:id', (req, res) => {
    try {
        const data = readProducts();
        const product = data.products.find(p => p.id ===
parseInt(req.params.id));

        if (!product) {
            return res.status(404).json({ error: 'Product not found' });
        }

        res.json(product);
    } catch (error) {
        res.status(500).json({ error: 'Failed to fetch product' });
    }
});

// POST /products - Add new product
app.post('/products', (req, res) => {
    try {
        const data = readProducts();
        const newProduct = {
            id: Date.now(), // Simple ID generation
            ...req.body
        };

        data.products.push(newProduct);
        writeProducts(data);

        res.status(201).json(newProduct);
    } catch (error) {
        res.status(500).json({ error: 'Failed to add product' });
    }
});

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

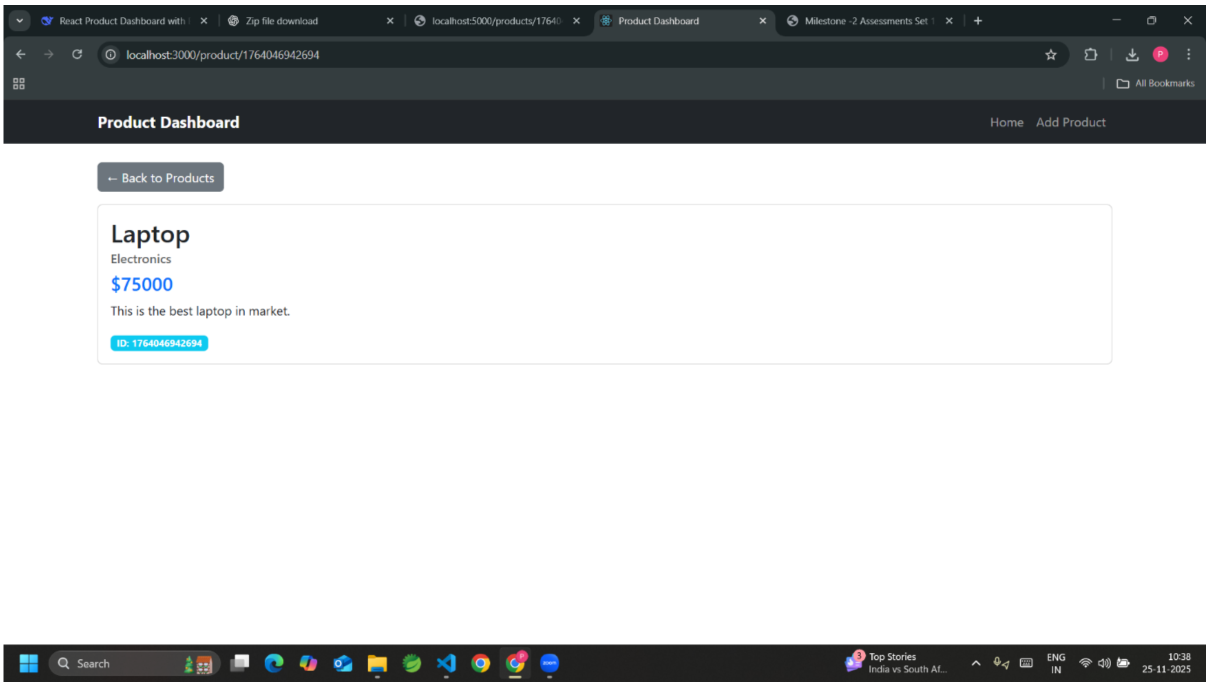
ProductDetail.js with API Integration

```
import React, { useState, useEffect } from 'react';
import { useParams, Link } from 'react-router-dom';
import axios from 'axios';

const ProductDetail = () => {
    const { id } = useParams();
    const [product, setProduct] = useState(null);
    const [loading, setLoading] = useState(true);
    const [error, setError] = useState(null);

    useEffect(() => {
        const fetchProduct = async () => {
            try {
                setLoading(true);
                const response = await
axios.get(`http://localhost:5000/products/${id}`);
                setProduct(response.data);
            } catch (err) {
                setError('Product not found');
            } finally {
                setLoading(false);
            }
        };

        fetchProduct();
    }, [id]);

    if (loading) {
        return (
            <div className="container mt-4">
                <div className="d-flex justify-content-center">
                    <div className="spinner-border" role="status">
                        <span className="visually-hidden">Loading...</span>
                    </div>
                </div>
            </div>
        );
    }
    if (error) {
        return (
            <div className="container mt-4">
                <div className="alert alert-danger" role="alert">
                    {error}
                </div>
                <Link to="/" className="btn btn-secondary">← Back to 
Products</Link>
            </div>
        );
    }

    return (
        <div className="container mt-4">
            <Link to="/" className="btn btn-secondary mb-3">← Back to 
Products</Link>

            {product && (
                <div className="card">
                    <div className="card-body">
                        <h2 className="card-title">{product.name}</h2>
                        <h6 className="card-subtitle mb-2 text-
muted">{product.category}</h6>
                        <p className="card-text h4 text-
primary">${product.price}</p>
                        <p className="card-text">{product.description}</p>
                        <div className="mt-3">
                            <span className="badge bg-info">ID: 
{product.id}</span>
                        </div>
                    </div>
                </div>
            )}
        </div>
    );
};

export default ProductDetail;
```

App.js Router Configuration

```
import React from 'react';
import { BrowserRouter as Router, Routes, Route, Link } from 'react-router-
dom';
import { ProductProvider } from './context/ProductContext';
import Home from './pages/Home';
import AddProduct from './pages/AddProduct';
import ProductDetail from './components/ProductDetail';
import 'bootstrap/dist/css/bootstrap.min.css';
import './App.css';

function App() {
    return (
        <ProductProvider>
            <Router>
                <nav className="navbar navbar-expand-lg navbar-dark bg-dark">
                    <div className="container">
                        <Link className="navbar-brand" to="/">Product
Dashboard</Link>
                        <div className="navbar-nav">
                            <Link className="nav-link" to="/">Home</Link>
                            <Link className="nav-link" to="/add-product">Add
Product</Link>
                        </div>
                    </div>
                </nav>

                <Routes>
                    <Route path="/" element={<Home />} />
                    <Route path="/add-product" element={<AddProduct />} />
                    <Route path="/product/:id" element={<ProductDetail />} />
                </Routes>
            </Router>
        </ProductProvider>
    );
}

export default App;
```

Output:



Product Dashboard

Laptop

Electronics

$75000

This is the best laptop in market.

ID: 1764046942694

Technical Achievements

- React Router DOM implementation
- Dynamic route parameters
- RESTful API endpoints (GET)
- Comprehensive error handling
- Loading state management
- Route transitions and navigation

User Story 3 - Add Product Form using Formik + Yup

Implementation Approach

1. **Form Management:** Formik for form state and handling
2. **Validation Schema:** Yup for client-side validation
3. **Global State:** Context API for product list updates
4. **API Integration:** POST request to backend
5. **User Experience:** Form validation feedback and success handling

Key Code Snippets

ProductContext.js (Context API)

```
import React, { createContext, useContext, useReducer } from 'react';

const ProductContext = createContext();
const productReducer = (state, action) => {
    switch (action.type) {
        case 'SET_PRODUCTS':
            return { ...state, products: action.payload };
        case 'ADD_PRODUCT':
            return { ...state, products: [...state.products, action.payload]
};
        default:
            return state;
    }
};

export const ProductProvider = ({ children }) => {
    const [state, dispatch] = useReducer(productReducer, { products: [] });

    return (
        <ProductContext.Provider value={{ state, dispatch }}>
            {children}
        </ProductContext.Provider>
    );
};

export const useProduct = () => {
    const context = useContext(ProductContext);
    if (!context) {
        throw new Error('useProduct must be used within a ProductProvider');
    }
    return context;
};
```

AddProduct.js (Formik + Yup)

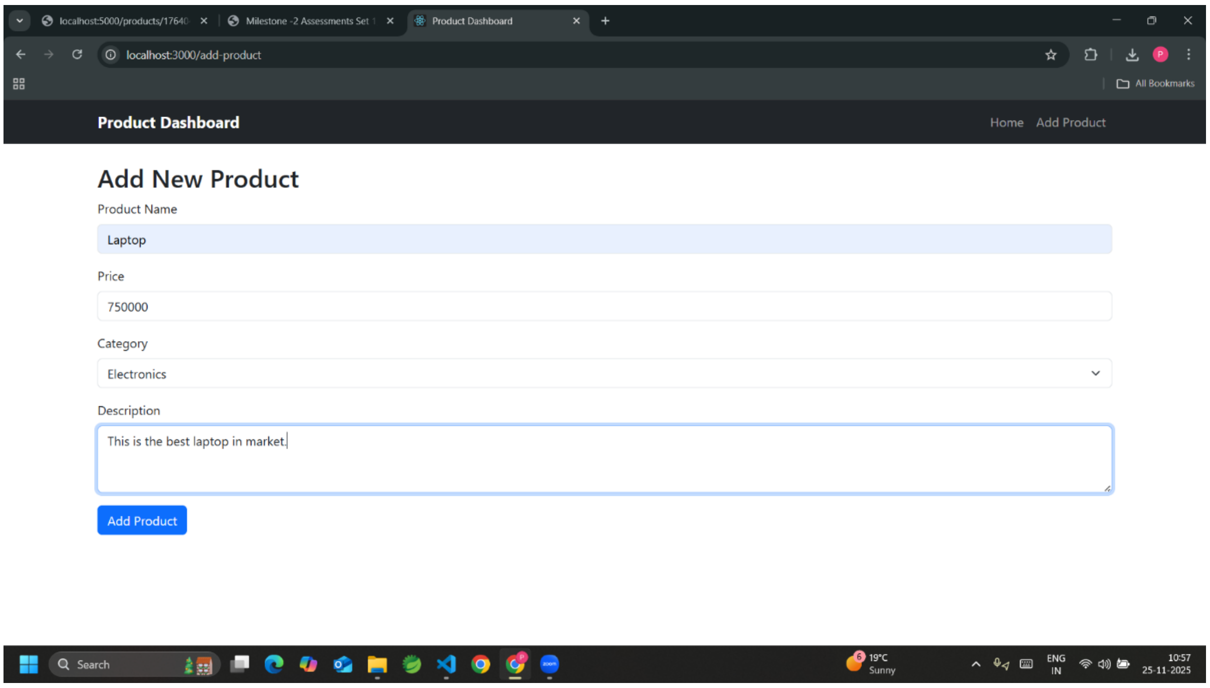
```
import React from 'react';
import { Formik, Form, Field, ErrorMessage } from 'formik';
import * as Yup from 'yup';
import axios from 'axios';
import { useProduct } from '../context/ProductContext';
import { useNavigate } from 'react-router-dom';

const AddProduct = () => {
    const { dispatch } = useProduct();
    const navigate = useNavigate();

    const validationSchema = Yup.object({
        name: Yup.string()
            .required('Product name is required')
            .min(2, 'Name must be at least 2 characters'),
        price: Yup.number()
            .required('Price is required')
            .positive('Price must be positive'),
        category: Yup.string()
            .required('Category is required'),
        description: Yup.string()
            .required('Description is required')
            .min(10, 'Description must be at least 10 characters')
    });

    const handleSubmit = async (values, { setSubmitting, resetForm }) => {
        try {
            const response = await
axios.post('http://localhost:5000/products', values);
            dispatch({ type: 'ADD_PRODUCT', payload: response.data });
            resetForm();
            navigate('/');
        } catch (error) {
            console.error('Failed to add product:', error);
        } finally {
            setSubmitting(false);
        }
    };

    return (
        <div className="container mt-4">
            <h2>Add New Product</h2>
            <Formik
                initialValues={{
                    name: '',
                    price: '',
                    category: '',
                    description: ''
                }}
                validationSchema={validationSchema}
                onSubmit={handleSubmit}
            >
                {({ isSubmitting }) => (
                    <Form>
                        <div className="mb-3">
                            <label htmlFor="name" className="form-
label">Product Name</label>
                            <Field type="text" name="name" className="form-
control" />
                            <ErrorMessage name="name" component="div"
className="text-danger" />
                        </div>
                        <div className="mb-3">
                            <label htmlFor="price" className="form-
label">Price</label>
                            <Field type="number" name="price" className="form-
control" step="0.01" />
                            <ErrorMessage name="price" component="div"
className="text-danger" />
                        </div>

                        <div className="mb-3">
                            <label htmlFor="category" className="form-
label">Category</label>
                            <Field as="select" name="category"
className="form-select">
                                <option value="">Select a category</option>
                                <option
value="Electronics">Electronics</option>
                                <option value="Home Appliances">Home
Appliances</option>
                                <option value="Clothing">Clothing</option>
                                <option value="Books">Books</option>
                            </Field>
                            <ErrorMessage name="category" component="div"
className="text-danger" />
                        </div>

                        <div className="mb-3">
                            <label htmlFor="description" className="form-
label">Description</label>
                            <Field as="textarea" name="description"
className="form-control" rows="3" />
                            <ErrorMessage name="description" component="div"
className="text-danger" />
                        </div>

                        <button
                            type="submit"
                            className="btn btn-primary"
                            disabled={isSubmitting}
                        >
                            {isSubmitting ? 'Adding...' : 'Add Product'}
                        </button>
                    </Form>
                )}
            </Formik>
        </div>
    );
};
```

```
export default AddProduct;
```

Backend POST Endpoint

```
// POST /products - Add new product
app.post('/products', (req, res) => {
    try {
        const data = readProducts();
        const newProduct = {
            id: Date.now(), // Simple ID generation
            ...req.body
        };

        data.products.push(newProduct);
        writeProducts(data);

        res.status(201).json(newProduct);
    } catch (error) {
        res.status(500).json({ error: 'Failed to add product' });
    }
});
```

Output:


localhost:3000/add-product

Product Dashboard | Home | Add Product

Add New Product

Product Name: Laptop

Price: 750000

Category: Electronics

Description: This is the best laptop in market.

Add Product

Technical Achievements

- Formik form state management
- Yup validation schema implementation

- Context API for global state
- POST API integration
- Form validation feedback
- Success handling and navigation
- Bootstrap form styling

Bonus Challenge - Lazy Loading

Implementation Approach

```
// In App.js
<Routes>
    <Route path="/product/:id" element={
        <Suspense fallback={<div>Loading...</div>}>
            <ProductDetail />
        </Suspense>
    } />
</Routes>
```

API Testing Results

Backend Endpoints Verified

- GET /products - Returns product array
- GET /products/1 - Returns single product
- POST /products - Successfully adds new product

Conclusion

All user stories have been successfully implemented with proper code structure, error handling, and user experience considerations. The application demonstrates comprehensive understanding of React concepts, API integration, and modern web development practices.