
AN INTERNSHIP REPORT

Name: Sanjaieswaran A
Reg.no: 920223243048
Title: My Store – Online E-Commerce Platform
Class: 3rd Artificial Intelligence and Data Science

Project Overview

This project is a full-stack **E-commerce Web Application** developed using **React.js, Express.js, Node.js, and MongoDB**. The application provides a seamless platform for both buyers and sellers. Users can register, log in, browse products, manage their shopping cart, and place orders. Sellers can manage their product listings through a dedicated seller dashboard.

The system ensures secure authentication using JWT and stores application data in MongoDB.

Key Features:

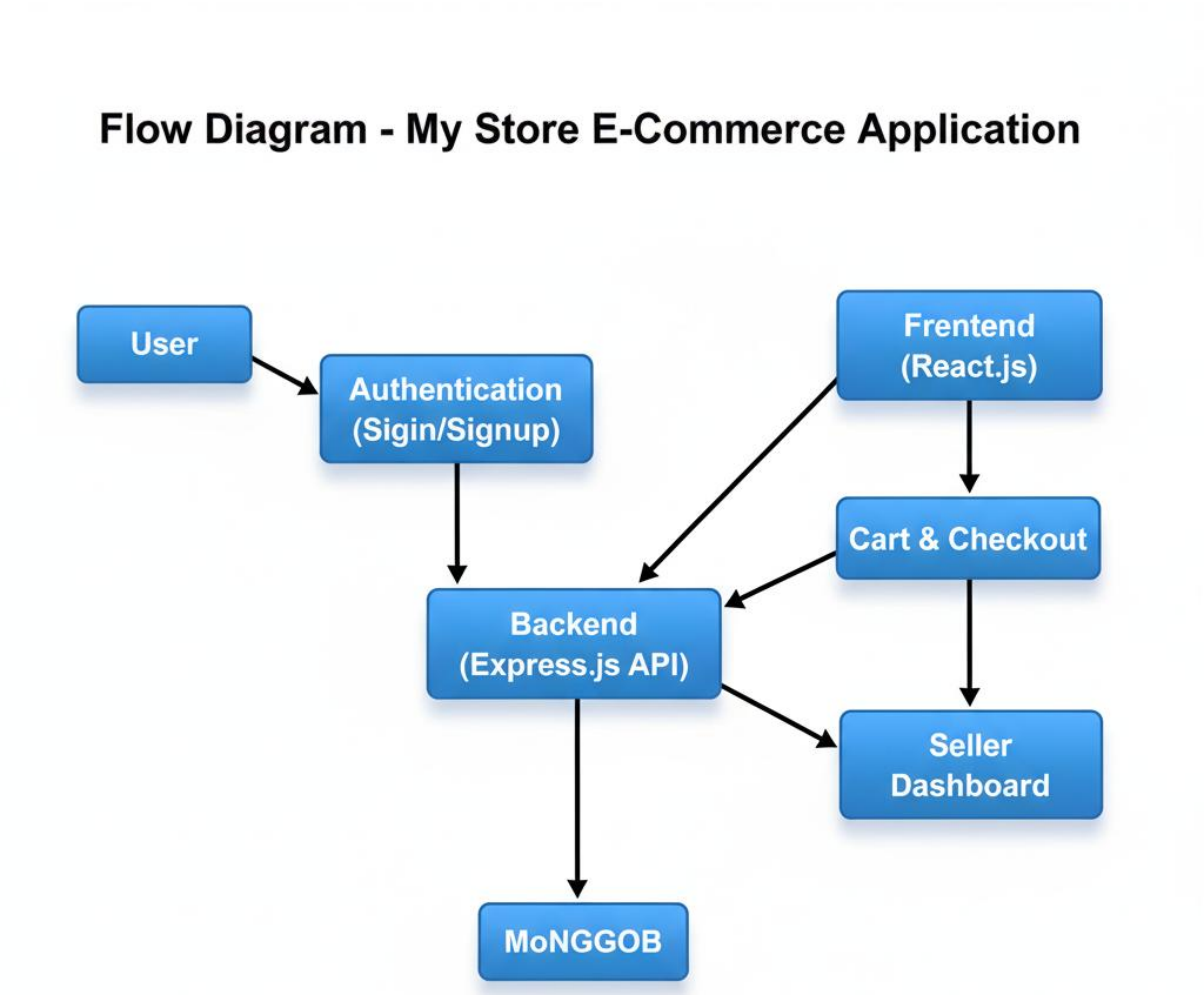
- User Registration and Login with JWT authentication.
- Role-based access (User / Seller).
- Product management: Add, Update, Delete (Seller only).
- Browse and filter products (Mobiles, Home Products).
- Shopping Cart with add/remove/update features.
- Checkout with shipping details.
- Place orders and store them in MongoDB.
- Contact form for customer support.
- Responsive, modern UI using CSS and React components.

Workflow Process

1. **UI Design** → Developed a modern user interface using **React.js and CSS**.
2. **Backend Development** → Built REST APIs with **Express.js** and connected them to MongoDB.
3. **Authentication** → Implemented secure login using **JWT and bcrypt.js**.

4. **Database Integration** → Stored users, products, carts, orders, and contacts in **MongoDB**.
5. **Frontend-Backend Connection** → Connected React frontend with backend APIs using fetch.
6. **Testing & Deployment** → Verified all modules (Auth, Cart, Orders, Contact) and ensured smooth flow.

Flow Diagram



Source Code (Express Backend)

// server.js

```
const express = require('express');
const mongoose = require('mongoose');
const cors = require('cors');
const bodyParser = require('body-parser');
require('dotenv').config();

const app = express();
app.use(cors());
app.use(bodyParser.json());

// Import Routes
const authRoutes = require('./routes/auth');
const cartRoutes = require('./routes/cart');
const orderRoutes = require('./routes/order');
const contactRoutes = require('./routes/contact');
const productRoutes = require('./routes/products');

// Use Routes
app.use('/api/auth', authRoutes);
app.use('/api/cart', cartRoutes);
app.use('/api/order', orderRoutes);
app.use('/api/contact', contactRoutes);
app.use('/api/products', productRoutes);

// Connect MongoDB
mongoose.connect(process.env.MONGO_URI)
  .then(() => console.log(" MongoDB Connected"))
  .catch(err => console.error(" MongoDB Error:", err));
```

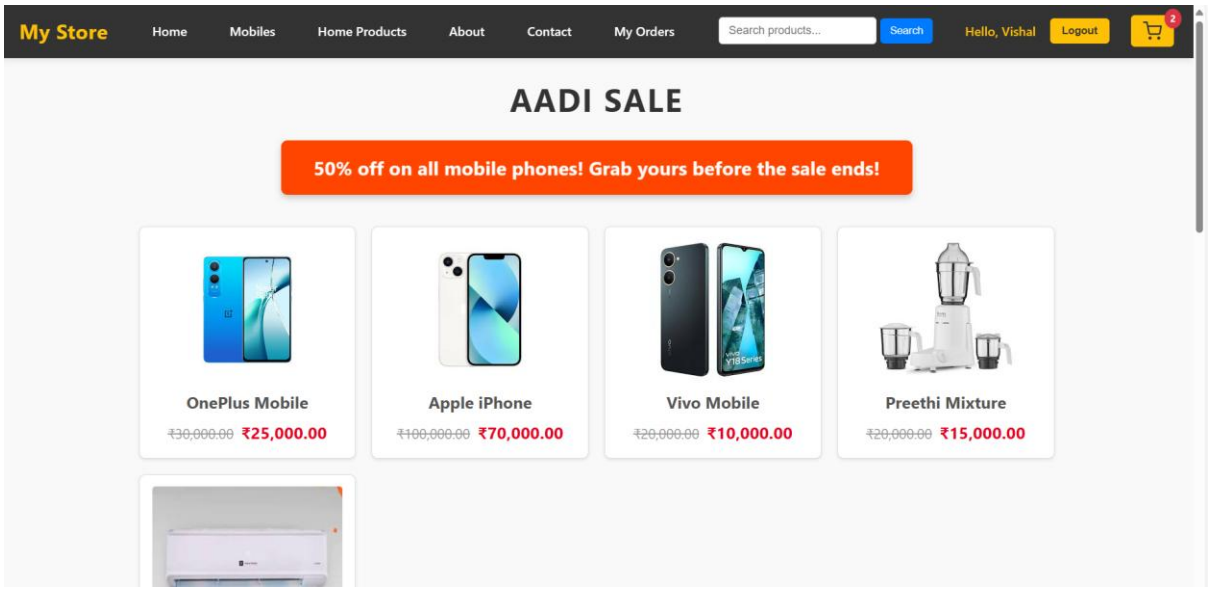
```
// Start Server
const PORT = process.env.PORT || 5000;
app.listen(PORT, () => {
  console.log(` Server running on http://localhost:${PORT}`);
});
```

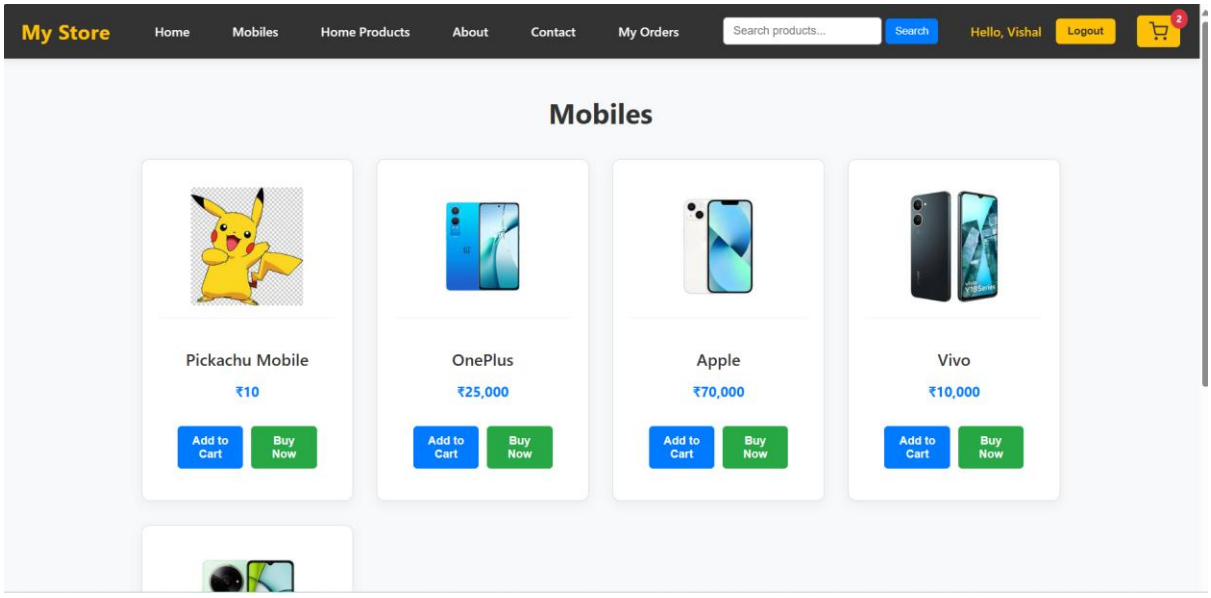
React.js Component (Cart)

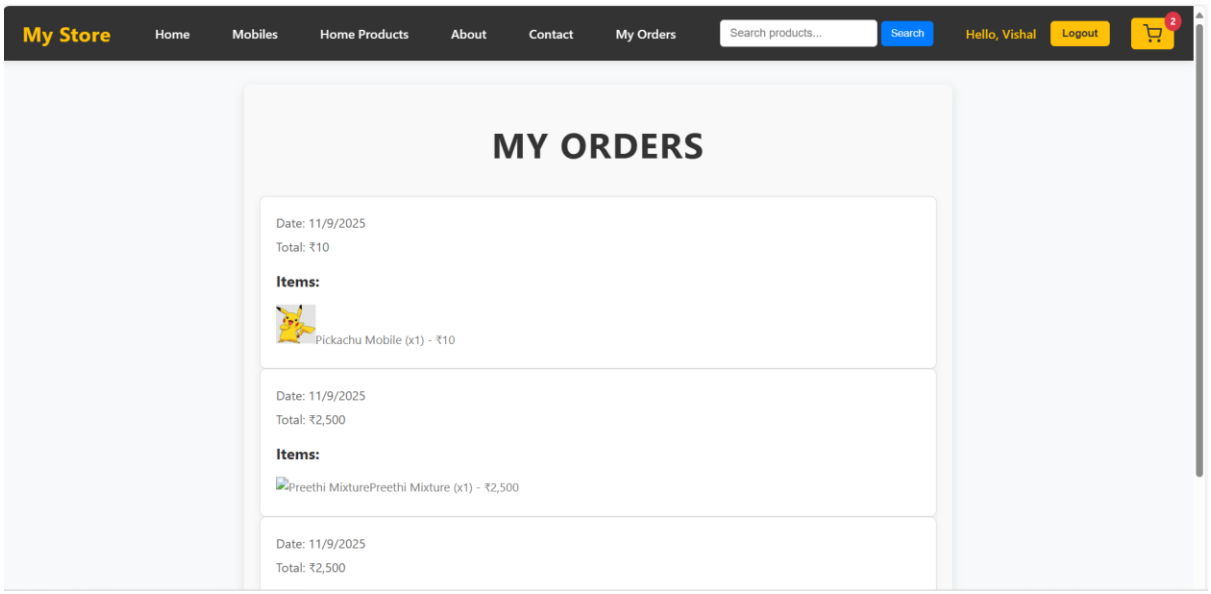
```
// cart.js
import React, { useState } from 'react';
import CartItem from './CartItem';
import CartSummary from './CartSummary';
import CheckoutForm from './components/CheckoutForm';
import './cart.css';

const Cart = ({ cartItems, UpdateQuantity, clearCart }) => {
  const [showCheckout, setShowCheckout] = useState(false);
  const totalItems = cartItems.reduce((sum, item) => sum + item.quantity, 0);
  const totalPrice = cartItems.reduce((sum, item) => sum + item.price * item.quantity, 0);

  return (
    <div className="cart-container">
      <h2>Your Cart</h2>
      {cartItems.length === 0 ? (
        <p>Cart is empty</p>
      ) : (
        <>
          {showCheckout ? (
```

```
          <CheckoutForm cart={cartItems} />
        ) : (
          <>
            <ul className="cart-list">
              {cartItems.map((item) => (
                <CartItem
                  key={item.id}
                  item={item}
                  UpdateQuantity={UpdateQuantity}
                />
              ))}
            </ul>
            <CartSummary
              totalItems={totalItems}
              totalPrice={totalPrice}
              handleBuyNow={() => setShowCheckout(true)}
            />
          </>
        )}
      </>
    )}
  </div>
 );
};

export default Cart;
```

Output:


My Store
Home
Mobiles
Home Products
About
Contact
My Orders
Search products...
Search
Hello, Vishal
Logout
2
AADI SALE
50% off on all mobile phones! Grab yours before the sale ends!
OnePlus Mobile
₹30,000.00 ₹25,000.00
Apple iPhone
₹100,000.00 ₹70,000.00
Vivo Mobile
₹20,000.00 ₹10,000.00
Preethi Mixture
₹20,000.00 ₹15,000.00


My Store
Home
Mobiles
Home Products
About
Contact
My Orders
Search products...
Search
Hello, Vishal
Logout
2
Mobiles
Pickachu Mobile
₹10
Add to Cart
Buy Now
OnePlus
₹25,000
Add to Cart
Buy Now
Apple
₹70,000
Add to Cart
Buy Now
Vivo
₹10,000
Add to Cart
Buy Now



Conclusion

The **My Store E-commerce Project** successfully demonstrates a complete **MERN stack application** with authentication, product management, shopping cart, and order placement.

It highlights the integration of **frontend (React.js)** with **backend APIs (Express.js)** and persistent storage in **MongoDB**. The application ensures secure login with JWT, supports role-based access for sellers, and provides a smooth checkout process.

Future Enhancements may include:

- Integration of a **payment gateway** (Stripe/Razorpay).
- Adding an **order history dashboard** for users.
- Implementing **search, filters, and product reviews**.
- Deploying the project to **Netlify/Vercel + MongoDB Atlas** for cloud accessibility.

This project demonstrates the essential skills required for building real-world, scalable web applications.
