

**VIETNAM NATIONAL UNIVERSITY HCMC
INTERNATIONAL UNIVERSITY**

**WEB APPLICATION
DEVELOPMENT
PROJECT
ONLINE SHOES STORE**

By

Pham Minh Thuc - ITITWE23033

Nguyen Huynh Dang Khoa - ITITWE22152

1. Introduction

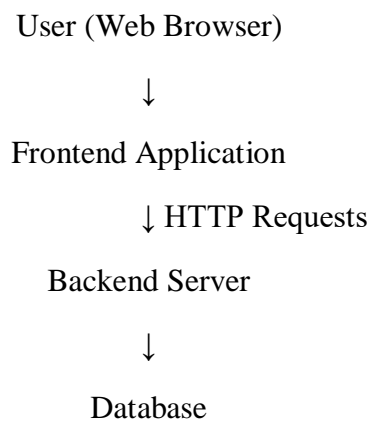
The purpose of the project is to design and implement an online shopping website where users can browse shoe products, create accounts, log in, and manage a shopping cart.

The application simulates a real life e-commerce website and demonstrates the application of modern web development concepts, including client–server interaction, user authentication, database integration, and responsive user interface design.

2. System Overview

The Shoes Store Web Application follows a client–server architecture. The front-end is responsible for displaying the user interface and handling user interactions. The back-end processes requests, applies business logic, and communicates with the database. The database stores user information, products, and cart data.

Overall System Flow:



3. Implementation

3.1 Frontend Implementation

3.1.1 Frontend Architecture

The frontend of the application is developed using React, following the Single Page Application (SPA) approach. Page navigation is handled dynamically without refreshing the browser, which improves performance and user experience.

State management is handled using Redux Toolkit, allowing shared data such as user login status and shopping cart information to be accessed across different components.

3.1.2 Frontend Features

1. User Interface

- Clean and modern layout
- Easy navigation using a navigation bar

2. User Authentication Pages

- Registration page for new users
- Login page for existing users
- Form validation and feedback messages

3. Product Browsing

- Display a list of shoe products
- Show product details including image, name, and price

4. Shopping Cart

- Add products to cart
- Update quantities
- Display total price

3.1.3 Frontend Work Flow Example – User Registration

Process explanation:

- The user opens the registration page and fills in the form.
- The form submission handler collects user input.
- An HTTP POST request is sent to the server.
- The server responds with success or error information.
- The frontend updates the UI accordingly.

3.2 Backend Implementation

3.2.1 Backend Architecture

The backend is implemented using Node.js and Express.js. It is structured using a layered approach to improve code readability and maintainability. Routes define API endpoints → Controllers handle incoming requests → Services process business logic → Models interact with the database → Middlewares handle authentication and error processing

3.2.2 Backend Features

1. User Authentication

- Secure registration and login
- Password encryption
- Token-based authentication

2. Product Management

- Retrieve product data from database
- Support adding and updating products (admin functionality)

3. Cart Management

- Cart linked to each user account
- Persistent cart data stored in database

4. Middleware Handling

- Authentication middleware protects restricted endpoints
- Error handling middleware ensures consistent responses

3.2.3 Backend Work Flow.

1. User Registration

Client → POST /api/auth/register

→ authRoute

→ AuthController.registerUser()

→ Validate name, username, password

→ newUser.save()

→ Send success response

2. User Login

Client POST /login

→ authRoute

→ AuthController.loginUser()

→ User.findOne and compare

→ generate JWT

→ Return token and user data

3. Product Listing

Client GET /products

→ productRoute

→ ProductController.getAllProducts()

→ Product.find()

→ Return product list

4. Add to Cart

Client PUT /cart/:userId

→ cartRoute

→ authenticateUser

→ CartController.updateCart()

→ Cart.findOneAndUpdate()

→ Return updated cart

5. View Cart

Client GET /cart/:userId

→ cartRoute

→ authenticateUser

→ CartController.getCartByUser()

→ Cart.findOne ()

→ Return cart data

6. Logout

User clicks Logout

→ Frontend deletes JWT from localStorage

→ Redux clears user state

→ User session ends

→ Redirect to homepage

7. Admin – Add New Product

Admin POST /admin/products

→ productRoute

→ authenticateUser

→ authorizeRoles("admin")

→ uploadToCloudinary middleware

→ ProductController.createProduct()

→ product.save()

→ Return success response

8. Admin – Update Product

Admin PUT /admin/products/:id

→ productRoute

→ authenticateUser

- authorizeRoles("admin")
- ProductController.updateProduct()
- Product.findByIdAndUpdate()
- Return updated product

9. Admin – Delete Product

Admin DELETE /admin/products/:id

- productRoute
- authenticateUser
- authorizeRoles("admin")
- ProductController.deleteProduct()
- Product.findByIdAndDelete()
- Return deletion confirmation

4. Conclusion

The Shoes Store Web Application successfully demonstrates the practical application of web application development concepts. The project integrates frontend user interfaces with backend services and database operations in a structured and secure manner.

Through this project, the development team gained hands-on experience in building a real-world web application, managing user authentication, handling data flow, and implementing responsive design. The system can be extended further with features such as online payment processing, order tracking, and admin dashboards.