Project Design Document: SS Geeks Store

1. Overview

SS Geeks Store is an e-commerce web application that allows users to browse products, manage a shopping cart, create wishlists, and authenticate via a secure login system. The project is built with a **Vue.js** front-end and a **Spring Boot** back-end with a PostgreSQL database.

2. Architecture

2.1 Front-end

- **Framework:** Vue.js 3
- **State Management:** Vuex
- **Routing:** Vue Router
- **HTTP Requests:** Axios
- **Styling:** Bootstrap, CSS

2.2 Back-end

- **Framework:** Spring Boot
- **Database:** PostgreSQL
- **Authentication:** JWT-based authentication
- **Data Access:** JDBC with DAO pattern
- **Security:** Spring Security

3. System Components

3.1 Front-End Components

3.1.1 Components Folder

- **`LoadingSpinner.vue`** **-** Displays a loading spinner during asynchronous operations.
- **`ProductCard.vue`** - Displays a product card with a flip animation.

3.1.2 Router Folder

- **`index.js`** - Manages routing and authentication checks.

3.1.3 Services Folder

- **`AuthService.js`** - Handles authentication API requests.
- **`CartService.js`** - Manages cart-related API requests.
- **`ProductService.js`** - Fetches product data.
- **`WishlistService.js`** - Handles wishlist API calls.

3.1.4 Store Folder

- **`index.js`** - Vuex store for managing authentication state.

3.1.5 Views Folder

- **`CartView.vue`** - Displays shopping cart contents.
- **`HomeView.vue`** - Homepage with featured products.
- **`LoginView.vue`** - User login page.
- **`LogoutView.vue`** - Logs out user and redirects to login.
- **`ProductDetailsView.vue`** - Displays details of a product.
- **`ProductsView.vue -`** Displays products with a search feature.
- **`ProfileView.vue`** - Displays user profile.
- **`RegisterView.vue`** - Handles user registration.
- **`WishlistView.vue`** - Manages user wishlists.

3.1.6 Main Entry Point

- **`App.vue`** - Main layout and navigation.
- **`main.js`** - Initializes the Vue app, Vuex store, Axios, and FontAwesome.

3.2 Back-End Components

3.2.1 Controllers

- **`AuthenticationController.java`** - Handles user authentication (login & registration).

- **`ProductController.java`** - Handles product management.
- **`WishlistController.java`** - Manages wishlists.
- **`CartController.java`** - Manages shopping cart actions.

3.2.2 Services

- **`CartService.java`** - Manages shopping cart operations such as adding/removing items, retrieving user carts, and clearing the cart.
- **`TaxService.java`** - Retrieves and calculates sales tax rates using an external tax API.
- **`WishlistService.java`** - Manages wishlist operations, including adding/removing items, retrieving wishlists, and validating product IDs.

3.2.3 Data Access Objects (DAOs)

- **`ProductDao.java`** - Interface for product-related database operations.
- **`CartItemDao.java`** - Interface for cart item operations.
- **`WishlistDao.java`** - Interface for wishlist operations.
- **`UserDao.java`** - Interface for user operations.
- **`JdbcWishlistDao.java`** - JDBC implementation for wishlist data access.
- **`JdbcUserDao.java` -** JDBC implementation for user data access.
- **`JdbcProductDao.java`** - JDBC implementation for product data access.
- **`JdbcCartItemDao.java`** - JDBC implementation for cart item data access.

3.2.4 Models

- **`Product.java`** - Represents a product.
- **`CartItem.java`** - Represents an item in a cart.
- **`Cart.java` -** Represents a shopping cart.
- **`Wishlist.java`** - Represents a wishlist.
- **`WishlistItem.java`** - Represents an item in a wishlist.
- **`User.java`** - Represents a user.
- **`RegisterUserDto.java`** – Represents a user being registered.
- **`Authority.java`** - Represents user roles.
- **`LoginDto.java`** - Represents login credentials sent from the client.
- **`LoginResponseDto.java`** - Represents authentication response containing a token and user details.

- **`TaxResponseDto.java`** - Represents tax-related information returned from the server.

3.3 Database Schema

The PostgreSQL database contains the following tables:

- **`users`** - Stores user details.
- **`product`** - Stores product information.
- **`wishlist`** - Stores user wishlists.
- **`wishlist_item`** - Stores items in a wishlist.
- **`cart_item`** - Stores shopping cart items.

4. User Authentication & Authorization

- Uses **JWT-based authentication** to secure API endpoints.
- **Vuex state management** stores authentication tokens.

5. Features

5.1 User Features

☑ Browse products ☑ Add/remove products from cart ☑ Create and manage wishlists ☑ Secure login and registration ☑ View profile details ☑ Search products

5.2 Admin Features (Future Enhancements)

◇ Product management (CRUD operations) ◇ User management ◇ Order processing

6. Conclusion

This design document outlines the structure, architecture, and features of the SS Geeks Store. Future improvements could include admin capabilities, order processing, and payment integration.