**Introduction to the Project**

* **What was your team project?**
  + Our project, **TheTechHeads**, is an e-commerce web application that focuses on selling tech-related footwear, with a primary emphasis on Nike shoes. The platform allows users to browse a wide range of products, filter by categories such as men, women, or kids, and purchase their favorite items. The system supports user authentication, shopping cart functionality, and payment processing, providing a full end-to-end shopping experience.
* **Motivation for the Project:**
  + E-commerce is at the heart of today’s digital economy. Our project explores building a seamless shopping experience that prioritizes ease of use, performance, and security. Through this project, we wanted to gain hands-on experience with popular web development tools, explore the nuances of back-end infrastructure, and learn how to integrate front-end and back-end systems. Given the widespread use of online shopping platforms, this project was not only technically rewarding but also highly relevant in today’s tech-driven landscape.

**2. Requirements**

* **Functional Requirements:**
  + Our project aimed to achieve the following functional goals:
    1. **User Authentication**: Users must be able to create an account, log in, and manage their profiles.
    2. **Product Browsing**: The platform allows users to browse and filter products by categories, such as Men, Women, and Kids.
    3. **Cart and Checkout**: Users should be able to add items to their cart, modify quantities, and proceed to checkout.
    4. **Payment Processing**: The system should securely process payments, validating the user’s card information and generating an order.
* **Performance Requirements:**
  + The system should load quickly and handle multiple users at once without degradation in performance.
  + The cart system should be responsive and capable of adding/removing items with minimal delay.
  + The back-end should handle data efficiently, minimizing response times for querying user or product data.

**3. Analysis and Design**

* **Major Classes Implemented (UML Class Diagram):**
  + Our architecture follows the **Model-View-Controller (MVC)** pattern, separating concerns between the business logic (Model), user interface (View), and control flow (Controller).

The key classes and their relationships are:

* + **User**: Represents a registered user in the system. It contains fields for personal data such as username, email, password, and role (admin or customer).
  + **Product**: Contains information about each product, including product name, price, description, and category (men, women, kids).
  + **Order**: Manages user orders by linking products to users. Each order includes the user’s details, the products purchased, quantities, and the total price.
  + **Cart**: Represents a temporary storage space for items before purchase. It holds product information and quantities before being converted into an order.
  + **Payment**: Handles the payment process by validating credit card details, ensuring the correct format for the card number, expiration date, and CVV.
* **Data Structures:**
  + **ArrayLists**: Used to store collections of products in the cart.
  + **HashMaps**: Used to map product IDs to quantities in the cart.
* **Algorithms:**
  + For the shopping cart, we implemented logic to dynamically calculate the total price based on the quantity and price of items in the cart. This also updates in real-time as users modify their cart contents.

**4. Database Schema**

* Our database schema ensures efficient storage and retrieval of data. The key tables are:
  1. **Users**: Stores information about each user, including their username, password, email, and role.
  2. **Products**: Stores details about each product, including its name, price, description, and category.
  3. **Orders**: Records user purchases, linking user accounts to the products they’ve bought.
  4. **Cart**: Temporarily holds the items a user wants to purchase.

**User Interface (UI)**

* **GUI Components:**
  + We designed a modern, responsive interface using **HTML**, **CSS**, and **Bootstrap**. The key components are:
    - **Product Grid**: Displays available products with images, names, prices, and descriptions.
    - **Add to Cart Button**: Allows users to add products to their cart.
    - **Checkout Page**: Summarizes the items in the cart and allows users to enter payment information.
* **Dynamic Interaction:**
  + **JavaScript**: Handles dynamic interactions like adding products to the cart and calculating the total in real-time.
  + **Form Validation**: Ensures that users input valid data when signing up or entering payment details.

**6. Implementation**

* **Tools and Technologies:**
  + **Front-end**: HTML, CSS, Bootstrap, JavaScript.
  + **Back-end**: Spring Boot with Java, handling user authentication, product management, and payment processing.
  + **Database**: MySQL is used to store user data, product information, and order history.
  + **Maven**: Used for managing dependencies, building the project, and ensuring a modular structure.
* **Testing and Validation:**
  + **JUnit** was used for unit testing the back-end logic. We tested essential components such as the shopping cart’s functionality, product addition, and user authentication.
  + The website was also manually tested to ensure it functions correctly across different devices and browsers.

**7. Accomplishments and Challenges**

* **Accomplishments:**
  + One of our main accomplishments is the seamless integration between the front-end and back-end, allowing users to interact with the system in real-time (e.g., adding items to the cart and updating totals).
  + Another highlight is the user-friendly payment validation system, which ensures correct formats for name, card number, expiry date, and CVV.
* **Challenges:**
  + Synchronizing data between the front-end and back-end posed a challenge. For example, ensuring that items in the cart are correctly updated on both the front-end (UI) and back-end (database).
  + Implementing responsive design across devices also required attention, especially for making the product grid adaptable to various screen sizes.
* **What We Learned:**
  + We learned a lot about handling back-end infrastructure, including user authentication, session management, and database integration. Additionally, ensuring security in payment processing taught us the importance of validating user inputs at every step.
* **Future Improvements:**
  + In the future, we would consider implementing more advanced features like dynamic pricing based on stock levels and adding user reviews. We’d also enhance security by incorporating token-based authentication for better session management.

**8. Conclusion**

* Our project, **TheTechHeads**, provided us with valuable experience in building a full-stack web application, from designing a user-friendly front-end to creating a robust back-end system.
* We’re proud of the smooth user experience we created, especially the cart and checkout process, and the lessons we learned about software development will undoubtedly benefit future projects.