**Week 1: E Commerce Project:**

**Video 1:**

* Understanding Project Development Lifecycle for E-Commerce Application using Angular & Spring Boot

**Concepts Covered:**

* Introduction to project development lifecycle
* Importance of structured project management from ideation to deployment
* Essential development tools: VS Code, Postman, IntelliJ, Java 8, Angular CLI, Node.js, MySQL
* Target audience: students, fresh graduates, and professionals
* Emphasis on hands-on learning and practical application development
* Importance of project management and roadmap planning

**How I Implemented These Concepts in My Project:**

* After watching this video, I gained a clearer understanding of how to systematically approach a project from scratch. To put this into practice, I started an e-commerce project using Angular and Spring Boot while focusing on:
* Setting up the development environment by installing and configuring necessary tools such as VS Code for frontend development, IntelliJ for backend, Postman for API testing, and MySQL Workbench for database management.
* Defining a structured project roadmap, breaking the development into manageable phases: Requirement Analysis, Database Design, Backend API Development, Frontend UI Implementation, Testing, and Deployment.
* Utilizing project management techniques like Agile methodology, setting milestones for each development phase.
* Building a foundation for modular development, ensuring backend (Spring Boot) and frontend (Angular) are properly structured for scalability.

**Challenges Faced and How I Resolved Them:**

* Challenge: Setting up the project structure and configuring dependencies correctly
* Solution: Referred to the video’s suggested setup, followed best practices for Angular & Spring Boot integration, and debugged dependency issues by resolving conflicts in package management (Maven for Spring Boot and npm for Angular).
* Challenge: Understanding the project development lifecycle and how to manage it effectively
* Solution: Implemented a task breakdown structure using tools like Jira and Trello to track progress and assigned clear milestones.

**Video 2:**

### **Setting Up Front-End and Back-End for E-Commerce Application**

#### **Concepts Covered:**

* Setting up **Angular and Spring Boot** projects from scratch
* **JWT authentication and role-based access control** for user management
* **Environment configuration**: cloning repositories, database setup, and port configuration

#### **How I Implemented These Concepts in My Project:**

* Cloned the **GitHub repositories** and configured **MySQL database** for seamless integration
* Implemented **JWT-based authentication** with user roles (Admin/User) to secure API access
* Ensured proper **port mapping and environment setup** for front-end and back-end communication

#### **Challenges Faced and How I Resolved Them:**

* **Dependency issues while setting up Angular and Spring Boot**

Resolved by checking package versions and updating dependencies accordingly

* **JWT authentication errors during login implementation**

Debugged by verifying token generation, storage, and API authorization logic

* **Database connection issues** due to incorrect schema configurations

Fixed by properly setting up database properties and ensuring MySQL service was running

This session helped solidify my understanding of **full-stack project setup**, authentication, and security best practices, which I can now efficiently apply in real-world applications.

**Video 3:**

### **Renaming a Spring Boot Project for Better Clarity and Organization**

#### **Concepts Covered:**

* **Project Naming Conventions:** Importance of meaningful project names for clarity and maintainability.
* **Step-by-Step Renaming Process:** Updating folder names, package names, and configuration files (pom.xml).
* **Refactoring and Maven Configuration:** Ensuring all references are updated for a smooth transition.

#### **How I Implemented These Concepts in My Project:**

* Renamed my **Spring Boot project** from a generic name to one that better reflects the application’s purpose.
* Updated **package structures, main class, and pom.xml settings** to ensure consistency.
* Used **IntelliJ's refactoring tools** to rename project files while maintaining project integrity.

#### **Challenges Faced and How I Resolved Them:**

* **Issue:** Errors in Maven dependencies after renaming project files.

**Solution:** Rebuilt the project and reloaded dependencies to reflect the new changes.

* **Issue:** Hardcoded references to the old package names causing build failures.

**Solution:** Used **Find & Replace** to update all occurrences of the old package name.

* **Issue:** Ensuring the project still runs correctly post-renaming.

**Solution:** Verified all changes, updated imports, and successfully restarted the application.

This session reinforced **best practices in project organization**, ensuring that my project remains scalable and easy to understand for future development.

**Video 4:**

### **Renaming an Angular Project for Consistency in E-Commerce Application**

#### **Concepts Covered:**

* **Consistent Naming Conventions:** Ensuring uniform project names across Angular and Spring Boot for better maintainability.
* **Updating Configuration Files:** Modifying package.json, package-lock.json, and other settings post-renaming.
* **Testing Post-Renaming:** Verifying seamless integration and JWT authentication for admin and user roles.

#### **How I Implemented These Concepts in My Project:**

* Renamed the **Angular project folder and updated project references** to align with backend naming.
* Updated **configuration files (package.json, angular.json)** to reflect the new project name.
* **Tested login functionality** to ensure JWT authentication and role-based access control worked correctly.

#### **Challenges Faced and How I Resolved Them:**

* **Errors due to outdated project name references** in package.json and angular.json.

**Solution:** Manually updated all occurrences and rebuilt the project to apply changes.

* **Login issues post-renaming due to mismatched API calls.**

**Solution:** Verified API endpoints and updated base URLs where necessary.

* **Ensuring backend and frontend integration remained intact.**

**Solution:** Ran tests and debugged network requests to confirm proper data flow.

This session reinforced **best practices in project organization, frontend-backend consistency, and systematic debugging**, making my development process more structured and efficient.

**Video 5:**

### **Integrating Angular Material UI in E-Commerce Application**

#### **Concepts Covered:**

* **Angular Material Installation & Setup:** Transitioning from Bootstrap to Angular Material for UI components.
* **Component Implementation:** Replacing Bootstrap navigation bar with Angular Material's toolbar.
* **UI Styling & Responsive Design:** Ensuring seamless user experience across different devices.

#### **How I Implemented These Concepts in My Project:**

* Installed **Angular Material** using CLI commands and configured themes for a modern UI.
* Replaced **Bootstrap navigation bar** with Angular Material’s toolbar and buttons for consistency.
* Integrated **login/logout functionality** with Angular Material components to enhance user interaction.

#### **Challenges Faced and How I Resolved Them:**

* **Errors in Angular Material module imports.**

**Solution:** Ensured proper import of required Material modules in app.module.ts.

* **UI inconsistencies when switching from Bootstrap.**

**Solution:** Adjusted CSS styles and spacing to align with Material Design principles.

* **Application failing to load after installation.**

**Solution:** Restarted the Angular server to apply new imports successfully.

This session provided hands-on experience in **modern UI development**, improving both usability and aesthetics in my e-commerce project.

**Video 6:**

### **Enhancing the Login Form with Angular Material in E-Commerce Application**

#### **Concepts Covered:**

* **Transition from Bootstrap to Angular Material** for improved UI consistency.
* **Module Management:** Importing MatFormFieldModule and MatInputModule to handle input fields.
* **Routing Enhancements:** Ensuring users are redirected to the home page after login.

#### **How I Implemented These Concepts in My Project:**

* Replaced the existing **Bootstrap-based login form** with Angular Material components.
* Updated **password field security** by setting the input type to ‘password’ for better data protection.
* Integrated **Angular Material buttons** for a modern and responsive UI design.

#### **Challenges Faced and How I Resolved Them:**

* **Errors due to missing Angular Material imports.**
  + **Solution:** Imported required modules into app.module.ts and ensured they were registered properly.
* **Misalignment of UI components post-transition.**
  + **Solution:** Adjusted CSS styles and utilized Bootstrap utility classes for layout consistency.
* **Routing issues after login/logout implementation.**
  + **Solution:** Updated the **router settings** to properly navigate users based on authentication state.

This session improved my **UI design approach**, making the login experience more user-friendly while maintaining security and functionality.