Final Presentation

App Component

- React application structure with components and routing.
- Uses React Router for navigation.
- State management using useState hook.
- Integration with a context provider (ShopContextProvider).
- Pages include Home, Shop, Contact, Cart, Login, Register, and AddProductForm.

FeedbackForm Component

- React component for a feedback form.
- Form submission handled by FormsPree service.
- Utilizes the textarea element for user input.
- Submission button triggers the Submit Feedback action.
- Clean and straightforward design.

Firebase Authentication in Login Component

- React component for user login.
- Utilizes Firebase Authentication service for login functionality.
- useState for managing email and password state.
- signInWithEmailAndPassword for user authentication.
- Successful login triggers a route change to '/shop'.
- Integration with Firebase for secure authentication.

Product Class

- Java class representing a product entity.
- Annotated with @Entity for JPA integration.
- Attributes include id, name, price, and image.
- Default and parameterized constructors.
- Getters and setters for attribute access.
- Designed for use with a JPA repository.

ProductController Class

- Spring Boot REST controller for product management.
- Endpoints for CRUD operations: getAll, create, getById, update, delete.
- Autowired with a ProductService.
- Exception handling for not found cases.
- Supports Cross-Origin Resource Sharing (CORS).

ProductRepository Interface

- Spring Data JPA repository interface.
- Extends JpaRepository<Product, Long>.
- Inherits CRUD operations from JpaRepository.
- Designed for managing Product entities in the database.

ProductService Class

- Spring service class for product-related business logic.
- Autowired with a ProductRepository.
- Methods for getAll, create, getById, update, delete.
- Exception handling for not found cases.
- Encapsulates business logic for product operations.

LoggingAspect Class

- Spring AOP aspect for logging.
- Logs method execution for classes containing "Product" in their names.
- Aspect applied to methods with @After advice.
- Utilizes JoinPoint for method information.
- Provides a simple console log statement.

ExceptionHandlingAspect Class

- Spring AOP aspect for exception handling.
- Catches exceptions for deleteProduct and updateProduct in ProductService.
- Aspect applied to methods with @After advice.
- Attempts to find a product by ID and throws an exception if not found.
- Designed for pre-validation before product update or deletion.