GENERAL REQUIREMENTS FOR ALL PROJECTS

PRECONDITION: Mandatory passing of tests in Autocode according to the project description.

Core Requirements

It is necessary to develop a Spring Boot application with a Front-End part, in which it is necessary:

- **to use:**

1. Spring Data JPA
2. Spring Security
3. DTOs (Data Transfer Objects)
4. SQL Scripts for Initial Data

- **to implement**:

5. UI internalization (English and local language)
6. Data Validation
7. Exception Handling
8. Database Configuration (Embedded H2 or similar)
9. Unit Testing (Cover Services)
10. Logging (business logic events, errors, security events)
11. Order Management

Nice to Have (to implement):

1. Searching, Pagination & Sorting
2. Stateless authentication (JWT-based) to avoid session management issues
3. Use BCrypt for password hashing (BCryptPasswordEncoder)
4. Unit Testing (Cover Controllers)

TOOLS & RECOMMENDATIONS (ALL PROJECTS)

We suggest you use the following tools or frameworks to implement projects:

1. Lombok
2. ModelMapper
3. AOP (Aspect-Oriented Programming) for implementing logging
4. Thymeleaf or another template engine for the front-end part

---

➢ **Recommended Spring Security Implementations**

1. Authentication Strategies (any of)

- **Use In-Memory Authentication** *(For quick setup/testing – Already suggested in Appliance Store task.)*
- **Support Database-backed Authentication** *(Store users in DB via Spring Security & JPA.)*
- **Use OAuth2 for third-party authentication** *(Google, Facebook, etc.).*

2. Authorization & Access Control

- **Role-based access control (RBAC)** *(ADMIN, EMPLOYEE, CUSTOMER, etc.)*
- **Method-level security** *(@PreAuthorize, @PostAuthorize, @Secured annotations.)*
- **URL-based security** *(Define access rules in SecurityFilterChain.)*
- **Custom security expressions** *(e.g., Checking ownership of an entity.)*

3. Password Security

- **Enforce strong password policies** *(Min length, special characters, etc.)*
- **Allow password reset functionality** *(e.g., Email-based recovery.)*

4. Session & Token Management

- **Set session timeouts and limits** *(For non-JWT implementations.)*
- **Implement refresh tokens** *(For long-lived authentication sessions.)*

5. Security Best Practices

- **Enable CSRF protection** *(Except for stateless APIs.)*
- **Disable exposing stack traces & detailed error messages.**
- **Use HTTPS for secure communication.**
- **Limit failed login attempts** *(Prevent brute-force attacks.)*