Based on your query from the Overview wiki page, I've analyzed the e-commerce microservices system and identified several areas that need improvement and issues that should be addressed.

## Security Vulnerabilities

### Hardcoded Credentials and Sensitive Information

The system exposes sensitive information in configuration files. In the User Service, there's a hardcoded email address in the login notification logic [1](#0-0) . This should be moved to environment variables or configuration properties.

### Overly Restrictive Email Validation

The Cart Service enforces an unnecessarily restrictive email pattern that only allows Gmail and Yahoo domains [2](#0-1) . This business constraint limits user adoption and should be reconsidered or made configurable.

## Code Quality Issues

### Inconsistent Naming Conventions

There are several naming inconsistencies throughout the codebase:

- `CartResposneDtos` has a typo (should be `CartResponseDtos`) [3](#0-2)

- `userPostelCode` should be `userPostalCode` [4](#0-3)

### Redundant Security Configuration

The User Service has multiple security filter chains with overlapping configurations. There's a third security configuration class `PublicSystemConfigs` [5](#0-4) that duplicates some of the main security configuration, which could lead to conflicts.

## Architecture Concerns

### Tight Coupling in Service Communication

The Cart Service directly calls both User and Product services for validation [6](#0-5) . This creates tight coupling and potential cascading failures. Consider implementing circuit breakers or async validation patterns.

### Inconsistent Error Handling

Different services use different exception types for similar scenarios. For example, `UserNotExistsException` is used in multiple services but may not be consistently handled across the system.

## Database and Performance Issues

### Database Configuration Concerns

The Cart Service uses `ddl-auto=create` which will recreate tables on every restart, potentially causing data loss in production environments. This should be changed to `validate` or `update` for production deployments.

### Missing Transaction Management

The cart operations don't appear to have proper transaction boundaries, which could lead to data inconsistency if operations fail partway through.

## Documentation and Maintenance

### Incomplete API Documentation

The README shows example endpoints but lacks comprehensive API documentation with request/response schemas, error codes, and authentication requirements.

### Geographic Limitations

The delivery service is hardcoded to only support Ukraine and Europe [7](#0-6) , which significantly limits the system's scalability and should be made configurable.

## Recommendations

1. \*\*Security\*\*: Move sensitive configuration to environment variables and implement proper secrets management

2. \*\*Code Quality\*\*: Fix naming inconsistencies and consolidate redundant security configurations

3. \*\*Architecture\*\*: Implement circuit breakers and async patterns for service communication

4. \*\*Database\*\*: Use appropriate DDL settings for different environments

5. \*\*Documentation\*\*: Create comprehensive API documentation and deployment guides

6. \*\*Scalability\*\*: Make geographic constraints configurable rather than hardcoded

## Notes

The system shows good separation of concerns with microservices architecture and proper JWT-based authentication. The OAuth2 implementation is well-structured, but the multiple security configurations need consolidation. The email validation constraint appears to be a business decision but should be documented as such rather than appearing as a technical limitation.

Wiki pages you might want to explore:

- [Security Architecture (JustDoItAbhi-org/MyEcomProjectAllServices)](/wiki/JustDoItAbhi-org/MyEcomProjectAllServices#2.1)

- [Core Services (JustDoItAbhi-org/MyEcomProjectAllServices)](/wiki/JustDoItAbhi-org/MyEcomProjectAllServices#3)

- [Cart Service (JustDoItAbhi-org/MyEcomProjectAllServices)](/wiki/JustDoItAbhi-org/MyEcomProjectAllServices#3.2)