RefundTest Documentation

Overview

This suite exercises the refund-related endpoints in `OrderController`:

- **requestRefund**: `PUT /api/order/requestRefund/{orderId}`
- **getActiveRefundRequests**: `GET /api/order/refundRequests/active`
- **approveRefund**: `PUT /api/order/refund/approve/{requestId}`
- **rejectRefund**: `DELETE /api/order/refund/reject/{requestId}`

Test Class Structure

- **Frameworks**: JUnit 5, Mockito
- **Mocked services**: `UserService`, `CartService`, `PaymentService`, `OrderService`, `OrderHistoryService`
- **Controller under test**: `OrderController` (standalone MockMvc)

Test Cases

1. **requestRefund_ShouldReturnOkAndMessage**
   - *Arrange*: stub `orderService.requestRefundSingle(...)` → `200 OK, "Refund requested"`
   - *Act*: `PUT /api/order/requestRefund/{orderId}`
   - *Assert*: status 200 and response body `"Refund requested"`
2. **getActiveRefundRequests_ShouldReturnListAsJson**
   - *Arrange*: stub `orderService.getRefundRequestsByProcessed(false)` → one `RefundRequest` with a single `RefundItem`
   - *Act*: `GET /api/order/refundRequests/active`
   - *Assert*: status 200 and JSON path checks on `requestId`, `orderId`, `items[0].productId`, etc.
3. **approveRefund_ShouldReturnOkAndApprovalMessage**
   - *Arrange*: stub `orderService.approveRefund(requestId)` → `200 OK, "Approved"`
   - *Act*: `PUT /api/order/refund/approve/{requestId}`
   - *Assert*: status 200 and body `"Approved"`
4. **rejectRefund_ShouldReturnOkAndRejectionMessage**
   - *Arrange*: stub `orderService.rejectRefund(requestId)` → `200 OK, "Rejected"`
   - *Act*: `DELETE /api/order/refund/reject/{requestId}`
   - *Assert*: status 200 and body `"Rejected"`

Mocking Strategy

- **Standalone MockMvc** via `MockMvcBuilders.standaloneSetup(orderController)`
- **@Mock** all services, **@InjectMocks** the controller
- **when(...).thenReturn(...)** for each service call

- **jsonPath** and **content()** assertions