

## LAB 8

### I. WORKFLOW

#### 1. GET All Customers

GET /api/customers  
→ Controller: getAllCustomers()  
→ Service: findAll() → stream().map(convertToResponseDTO)  
→ Repository: findAll()  
→ Response: 200 OK + List<CustomerResponseDTO>

```
Status: 200 OK    Size: 1.27 KB    Time: 8 ms

Response Headers 5 Cookies Results Docs
1 [ 2 { 3   "id": 1, 4   "customerCode": "C001", 5   "fullName": "John Doe", 6   "email": "john.doe@example.com", 7   "phone": "+1-555-0101", 8   "address": "123 Main St, New York, NY 10001", 9   "status": "ACTIVE", 10  "createdAt": "2025-12-06T16:33:34" 11 }.
```

#### 2. GET Customer by ID

Status: 200 OK Size: 256 Bytes Time: 16 ms

Response Headers 5 Cookies Results Docs

```
1  {
2      "id": 3,
3      "customerCode": "C003",
4      "fullName": "Bob Johnson",
5      "email": "bob.johnson@example.com",
6      "phone": "+1-555-0103",
7      "address": "789 Pine Rd, Chicago, IL 60601",
8      "status": "ACTIVE",
9      "createdAt": "2025-12-06T16:33:34"
10 }
```

GET /api/customers/{id}

→ Controller: getCustomerById(@PathVariable Long id)  
→ Service: findById(id).orElseThrow(ResourceNotFoundException)  
→ Repository: findById(id)  
→ Response: 200 OK + CustomerResponseDTO | 404 Not Found

### 3. POST Create Customer

Status: 201 Created Size: 277 Bytes Time: 115 ms

Response Headers 5 Cookies Results Docs

```
1  {
2      "id": 7,
3      "customerCode": "C006",
4      "fullName": "David Miller Jr.",
5      "email": "david.miller.jr@example.com",
6      "phone": "+155502323107",
7      "address": "1000 Broadway, Seattle, WA 98101",
8      "status": "ACTIVE",
9      "createdAt": "2025-12-06T10:48:18.0689001"
10 }
```

POST /api/customers + JSON body

- Controller: createCustomer(@Valid @RequestBody CustomerRequestDTO)
- Validation: @NotBlank, @Email, @Pattern annotations
- Service:
  - Check duplicate (existsByCustomerCode, existsByEmail)
  - convertToEntity(requestDTO)
  - save(customer)
  - convertToResponseDTO(savedCustomer)
- Repository: save()
- Response: 201 Created + CustomerResponseDTO | 409 Conflict

## 4. PUT Update Customer

The screenshot shows a Postman request for a PUT operation. The URL is `http://localhost:8089/api/customers/7`. The **Body** tab is selected, and the **JSON** tab is active. The JSON content is:

```

1  {
2    "customerCode": "C006",
3    "fullName": "David Miller Jr.",
4    "email": "david.miller.jr@example.com",
5    "phone": "+155502323107",
6    "address": "1000 Broadway, Seattle, WA 98101"
7  }
  
```

The response section shows the following details:

Status: 200 OK    Size: 269 Bytes    Time: 10 ms

Response	Headers	Cookies	Results	Docs
<pre> 1  { 2    "id": 7, 3    "customerCode": "C006", 4    "fullName": "David Miller Jr.", 5    "email": "david.miller.jr@example.com", 6    "phone": "+155502323107", 7    "address": "1000 Broadway, Seattle, WA 98101", 8    "status": "ACTIVE",   </pre>			<input type="button" value="Copy"/>	

PUT /api/customers/{id} + JSON body

- Controller: updateCustomer(@PathVariable Long id, @Valid @RequestBody)
- Service:
  - findById(id).orElseThrow()
  - Check email uniqueness (nếu thay đổi)
  - Update fields
  - save()
- Repository: save()
- Response: 200 OK + CustomerResponseDTO | 404 Not Found

## 5. DELETE Customer

The screenshot shows a Postman request for a DELETE operation. The URL is `http://localhost:8089/api/customers/7`. The Body tab is selected, containing the following JSON:

```
1 {  
2     "customerCode": "C006",  
3     "fullName": "David Miller Jr.",  
4     "email": "david.miller.jr@example.com",  
5     "phone": "+155502323107",  
6     "address": "1000 Broadway, Seattle, WA 98101"  
7 }
```

The response status is 200 OK, size is 51 bytes, and time is 32 ms. The response body is:

```
1 {  
2     "message": "Customer deleted successfully"  
3 }
```

DELETE /api/customers/{id}

- Controller: deleteCustomer(@PathVariable Long id)
- Service: existsById(id) → deleteById(id)
- Repository: deleteById()
- Response: 200 OK + {"message": "Customer deleted successfully"} | 404

## 6. Search Customers

The screenshot shows a Postman request for a GET operation. The URL is `http://localhost:8089/api/customers/search?keyword=john`. The Body tab is selected, containing the following JSON:

```
1 {  
2     "customerCode": "C006",  
3     "fullName": "David Miller Jr.",  
4     "email": "david.miller.jr@example.com",  
5     "phone": "+155502323107",  
6     "address": "1000 Broadway, Seattle, WA 98101"  
7 }
```

The response status is 200 OK, size is 513 bytes, and time is 16 ms. The response body is:

```
1 [  
2     {  
3         "id": 1,  
4         "customerCode": "C001",  
5         "fullName": "John Doe",  
6         "email": "john.doe@example.com",  
7         "phone": "+1-555-0101",  
8         "address": "123 Main St, New York, NY 10001",  
9     }]
```

```
GET /api/customers/search?keyword=john
→ Controller: searchCustomers(@RequestParam String keyword)
→ Service: searchByKeyword(keyword)
→ Repository: @Query JPQL (search fullName, email, customerCode)
→ Response: 200 OK + List<CustomerResponseDTO>
```

## II. VALIDATION & EXCEPTION HANDLING

### Bean Validation (CustomerRequestDTO)

```
@NotBlank(message = "Customer code is required")
@Pattern(regexp = "^\w{3}\d{3}$", message = "Must start with C + numbers")
private String customerCode;

@email(message = "Invalid email format")
private String email;
```

### Global Exception Handler

```
@RestControllerAdvice
@ExceptionHandler(ResourceNotFoundException.class)
→ 404 Not Found + ErrorResponse JSON

@ExceptionHandler(MethodArgumentNotValidException.class)
→ 400 Bad Request + Map<field, errorMessage>

@ExceptionHandler(DuplicateResourceException.class)
→ 409 Conflict + ErrorResponse JSON
```