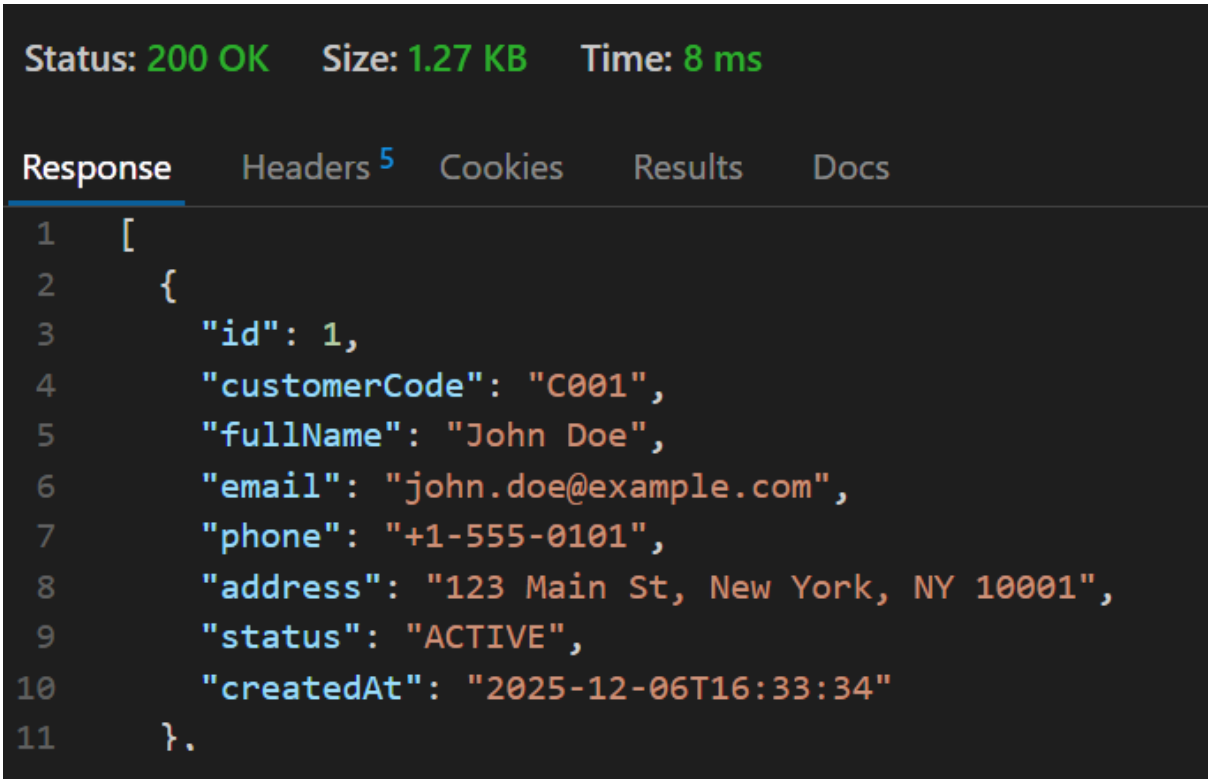
Trần Phúc Quý - ITITDK23027

LAB 8

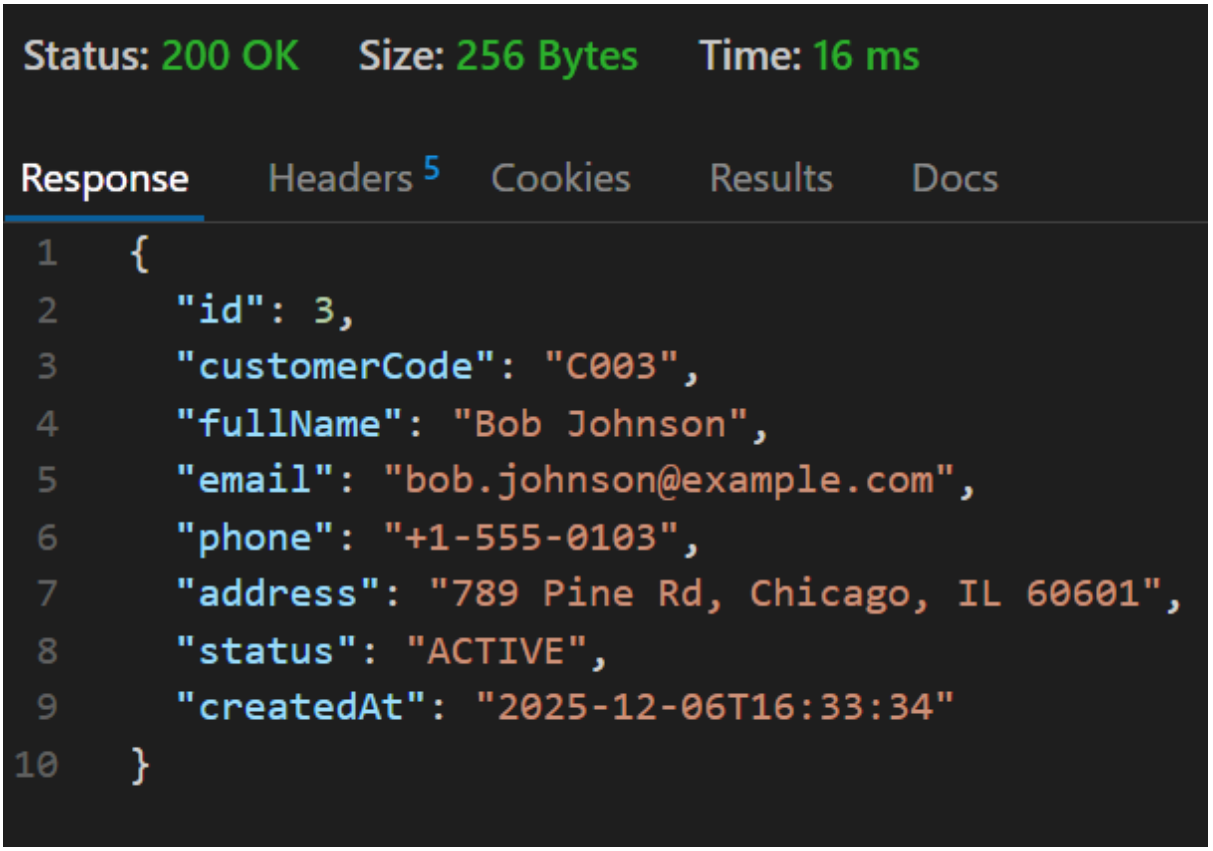
I. WORKFLOW

1. GET All Customers

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

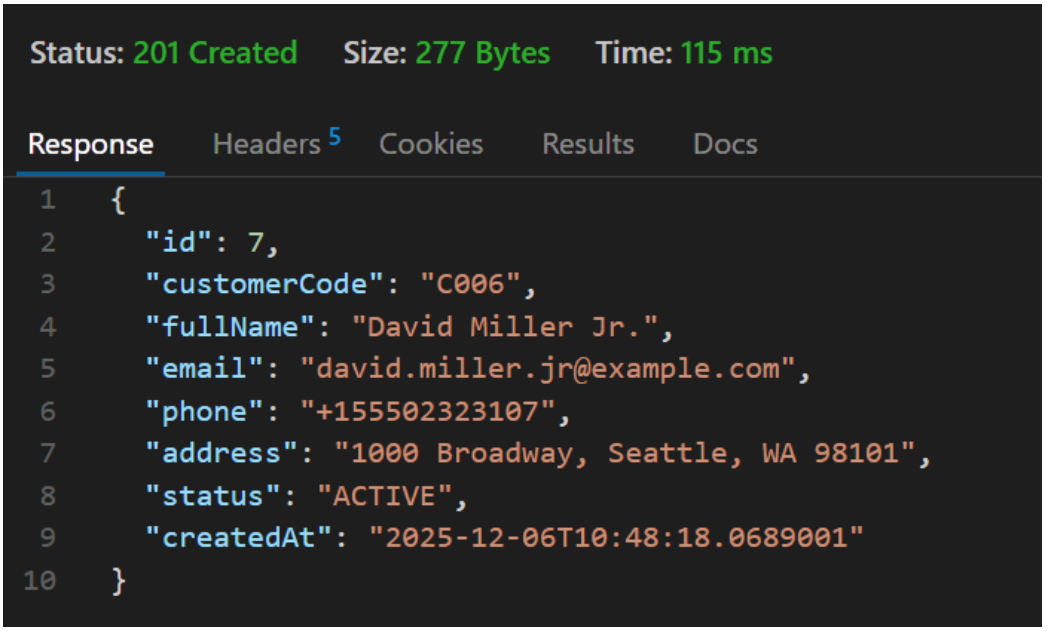


2. GET Customer by ID



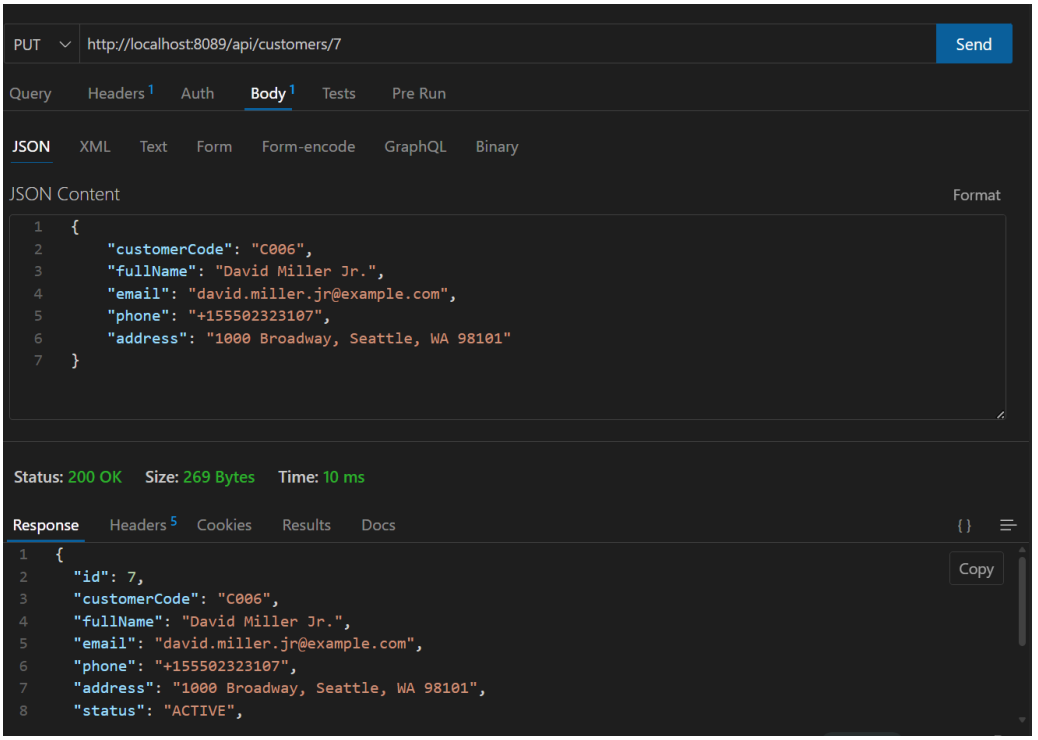
GET /api/customers/{id}
→ Controller: getCustomerById(@PathVariable Long id)
→ Service: findById(id).orElseThrow(ResourceNotFoundException)
→ Repository: findById(id)
→ Response: 200 OK + CustomerResponseDTO | 404 Not Found

3. POST Create Customer



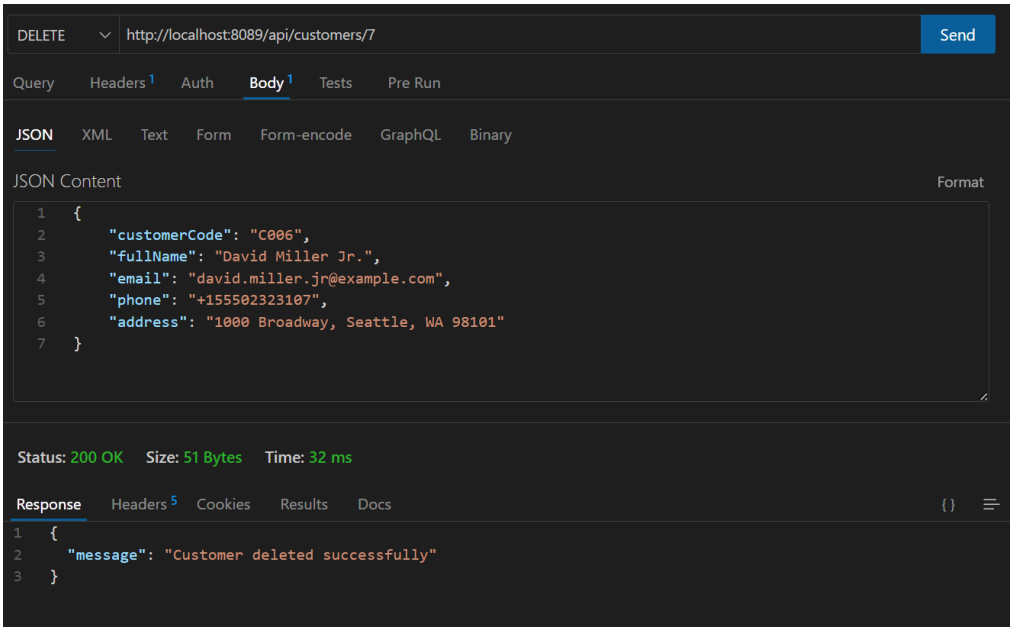
POST /api/customers + JSON body
→ Controller: createCustomer(@Valid @RequestBody CustomerRequestDTO)
→ Validation: @NotBlank, @Email, @Pattern annotations
→ Service:
  - Check duplicate (existsByCustomerCode, existsByEmail)
  - convertToEntity(requestDTO)
  - save(customer)
  - convertToResponseDTO(savedCustomer)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

4. PUT Update Customer



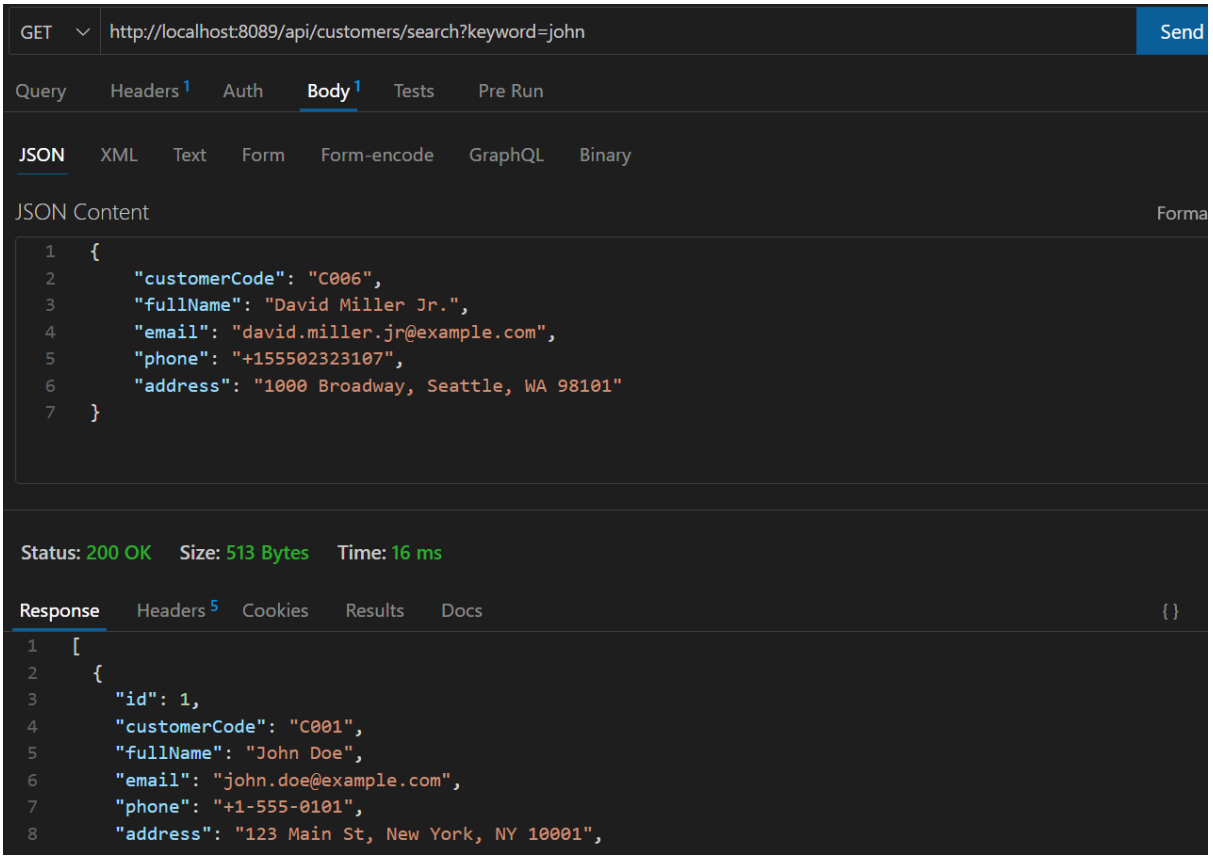
PUT /api/customers/{id} + JSON body
→ Controller: updateCustomer(@PathVariable Long id, @Valid @RequestBody)
→ Service:
  - findById(id).orElseThrow()
  - Check email uniqueness (nếu thay đổi)
  - Update fields
  - save()
→ Repository: save()
→ Response: 200 OK + CustomerResponseDTO | 404 Not Found

5. DELETE Customer



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



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 = "^C\\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
```