Experiment No.9

Date:05.05.25

Aim: Creating REST services: sending messages asynchronously

CO Mapping – CO 4

Objective:

1. To develop program based on REST services

Code:

```
// ================================================
// Main Application Class
// ================================================
package com.spring.cms;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

@SpringBootApplication
@EnableJpaRepositories
public class CmsApplication {
   public static void main(String[] args) {
      SpringApplication.run(CmsApplication.class, args);
   }
}

// ================================================
// Model Classes
// ================================================
package com.spring.cms.model;

import com.fasterxml.jackson.annotation.JsonProperty;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import org.springframework.stereotype.Component;

@Entity
```

```
public class Customer {
   @Id
   @GeneratedValue(strategy = GenerationType.AUTO)
   @JsonProperty("id")
   private int customerId;
   @JsonProperty("firstName")
   private String customerFirstName;
   @JsonProperty("lastName")
   private String customerLastName;
   @JsonProperty("email")
   private String customerEmail;

   public String getCustomerEmail() {
      return customerEmail;
   }

   public void setCustomerEmail(String customerEmail) {
      this.customerEmail = customerEmail;
   }

   public String getCustomerLastName() {
      return customerLastName;
   }

   public void setCustomerLastName(String customerLastName) {
      this.customerLastName = customerLastName;
   }

   public String getCustomerFirstName() {
      return customerFirstName;
   }

   public void setCustomerFirstName(String customerFirstName) {
      this.customerFirstName = customerFirstName;
   }

   public int getCustomerId() {
      return customerId;
   }

   public void setCustomerId(int customerId) {
      this.customerId = customerId;
   }
}
```

```
// ================================================
// DAO Classes
// ================================================
package com.spring.cms.dao;

import com.spring.cms.model.Customer;
import org.springframework.data.repository.CrudRepository;
import org.springframework.stereotype.Repository;

import java.util.List;

@Repository
public interface CustomerDAO extends CrudRepository<Customer,Integer> {
   @Override
   List<Customer> findAll();
}

// ================================================
// Service Classes
// ================================================
package com.spring.cms.service;

import com.spring.cms.dao.CustomerDAO;
import com.spring.cms.exception.CustomerNotFoundException;
import com.spring.cms.model.Customer;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;

import java.util.List;
import java.util.Optional;

@Component
public class CustomerService {
   @Autowired
   private CustomerDAO customerDAO;

   public Customer addCutomer(Customer customer){
      return customerDAO.save(customer);
   }

   public List<Customer> getCustomers(){
      return customerDAO.findAll();
   }
```

```
    public Customer getCustomer(int customerId){
        Optional<Customer> optionalCustomer = customerDAO.findById(customerId);
        if(optionalCustomer.isEmpty()){
            throw new CustomerNotFoundException("Customer Record not available");
        }
        return optionalCustomer.get();
    }

    public Customer updateCustomer(int customerId,Customer customer){
        customer.setCustomerId(customerId);
        return customerDAO.save(customer);
    }

    public void deleteCustomer(int customerId){
        customerDAO.deleteById(customerId);
    }
}

// ================================================
// API Classes
// ================================================
package com.spring.cms.api;

import com.spring.cms.model.Customer;
import com.spring.cms.service.CustomerService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@RequestMapping(value = "/customers")
public class CustomerResources {
    @Autowired
    private CustomerService customerService;

    @PostMapping
    public Customer addCustomer(@RequestBody Customer customer){
        return customerService.addCutomer(customer);
    }

    @GetMapping
    public List<Customer> getCustomers(){
```

```
        return customerService.getCustomers();
    }

    @GetMapping(value = "/{customerId}")
    public Customer getCustomer(@PathVariable("customerId") int customerId){
        return customerService.getCustomer(customerId);
    }

    @PutMapping(value = "/{customerId}")
    public Customer updateCustomer (@PathVariable("customerId") int
customerId,@RequestBody Customer customer){
        return customerService.updateCustomer(customerId,customer);
    }

    @DeleteMapping(value = "/{customerId}")
    public void deleteCustomer(@PathVariable("customerId") int customerId){
        customerService.deleteCustomer(customerId);
    }
}

package com.spring.cms.api;

import com.spring.cms.exception.ApplicationError;
import com.spring.cms.exception.CustomerNotFoundException;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.ControllerAdvice;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.context.request.WebRequest;
import
org.springframework.web.servlet.mvc.method.annotation.ResponseEntityExceptionHandler;

@ControllerAdvice
@RestController
public class errorHandler extends ResponseEntityExceptionHandler {
    @Value("${api_doc_url}")
    private String details;

    @ExceptionHandler(CustomerNotFoundException.class)
    public ResponseEntity<ApplicationError>
handlerCustomerNotFoundException(CustomerNotFoundException exception, WebRequest
webRequest){
```

```
        ApplicationError error = new ApplicationError();
        error.setCode(101);
        error.setMessage(error.getMessage());
        error.setDetails("");
        return new ResponseEntity<>(error, HttpStatus.NOT_FOUND);
    }
}

package com.spring.cms.api;

import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class HelloWorld {
    @RequestMapping(value = "/hello")
    public String sayHello(){
        return "Hello World";
    }
}

package com.spring.cms.api;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

import java.util.Date;

@RestController
public class Home {
    @GetMapping
    public String home(){
        return "Application is Working!! "+new Date();
    }
}

// ================================================
// Exception Classes
// ================================================
package com.spring.cms.exception;

public class ApplicationError {
    private int code;
    private String message;
```

```
    private String details;

    public String getDetails() {
        return details;
    }

    public void setDetails(String details) {
        this.details = details;
    }

    public String getMessage() {
        return message;
    }

    public void setMessage(String message) {
        this.message = message;
    }

    public int getCode() {
        return code;
    }

    public void setCode(int code) {
        this.code = code;
    }
}

package com.spring.cms.exception;

public class CustomerNotFoundException extends RuntimeException{
    public CustomerNotFoundException(String message){
        super(message);
    }
}

// ================================================
// Filter Classes
// ================================================
package com.spring.cms.filter;

import jakarta.servlet.*;
import org.springframework.core.annotation.Order;
import org.springframework.stereotype.Component;
```

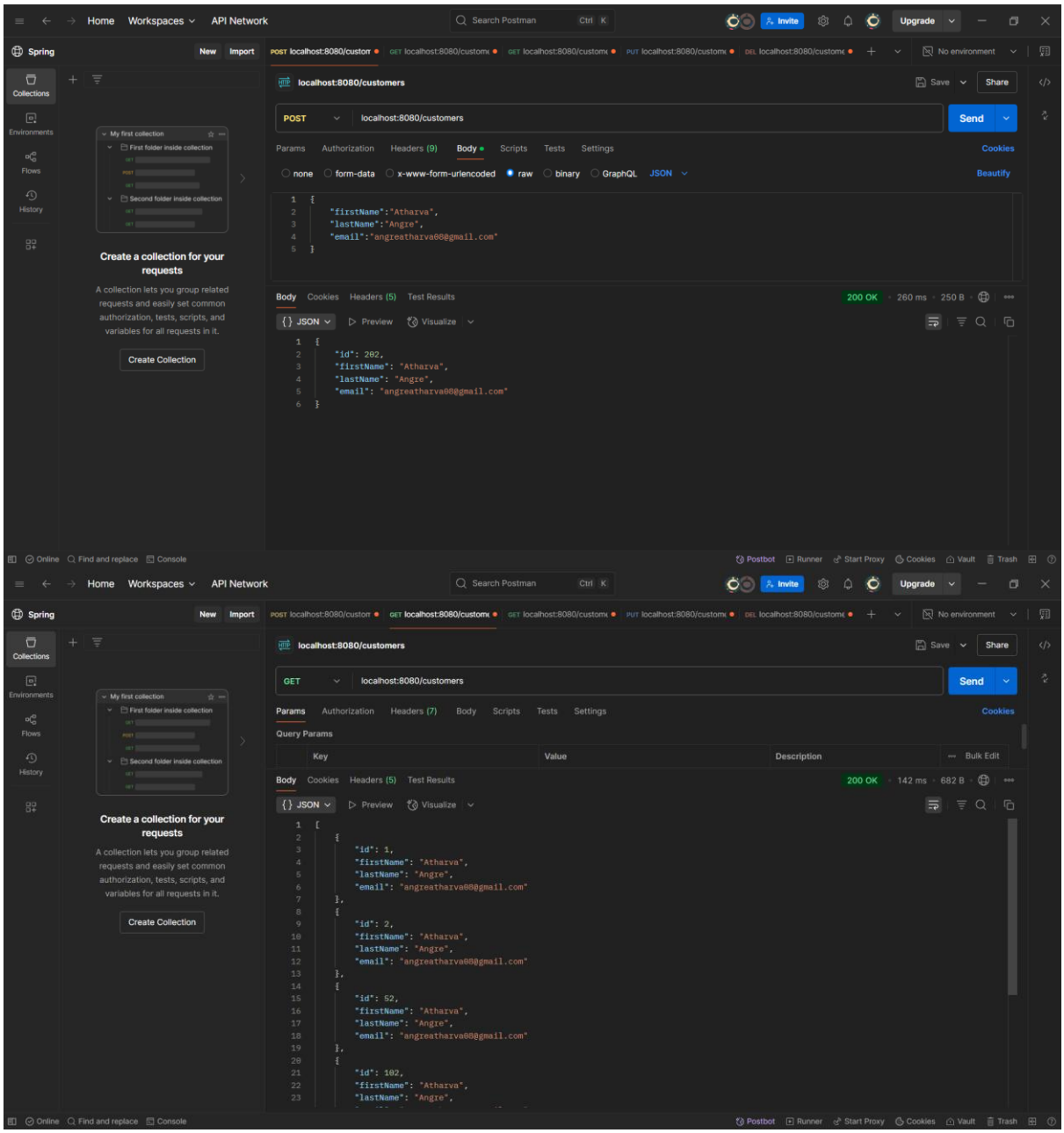
```
import java.io.IOException;

@Component
@Order(2)
public class MyFilter_1 implements Filter {
   @Override
   public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse,
FilterChain filterChain) throws IOException, ServletException {
      System.out.println("Filter 1 is Called...");
      filterChain.doFilter(servletRequest,servletResponse);
   }
}

package com.spring.cms.filter;

import jakarta.servlet.*;
import org.springframework.core.annotation.Order;
import org.springframework.stereotype.Component;

import java.io.IOException;

@Component
@Order(3)
public class MyFilter_2 implements Filter {
   @Override
   public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse,
FilterChain filterChain) throws IOException, ServletException {
      System.out.println("Filter 2 is Called...");
      filterChain.doFilter(servletRequest,servletResponse);
   }
}

package com.spring.cms.filter;

import jakarta.servlet.*;
import org.springframework.core.annotation.Order;
import org.springframework.stereotype.Component;

import java.io.IOException;

@Component
@Order(1)
public class MyFilter_3 implements Filter {
   @Override
```

```
   public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse,
FilterChain filterChain) throws IOException, ServletException {
      System.out.println("Filter 3 is Called...");
      filterChain.doFilter(servletRequest,servletResponse);
   }
}

package com.spring.cms.filter;

import jakarta.servlet.*;
import java.io.IOException;

public class MyNewFilter implements Filter {
   @Override
   public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse,
FilterChain filterChain) throws IOException, ServletException {
      System.out.println("The new Filter is called...");
      filterChain.doFilter(servletRequest,servletResponse);
   }
}

// ================================================
// Config Classes
// ================================================
package com.spring.cms.config;

import com.spring.cms.filter.MyNewFilter;
import org.springframework.boot.web.servlet.FilterRegistrationBean;
import org.springframework.context.annotation.Configuration;

@Configuration
public class MyFilterConfig {
   public FilterRegistrationBean<MyNewFilter> registrationBean(){
      FilterRegistrationBean<MyNewFilter> registrationBean = new
FilterRegistrationBean<>();
      registrationBean.setFilter(new MyNewFilter());
      registrationBean.addUrlPatterns("/customers/*");
      return registrationBean;
   }
}
```

Academic Year: 2024-25 **Semester: II** **Class: FYMCA**
Course Code: MC506 **Course Name: Advanced Java Programming**

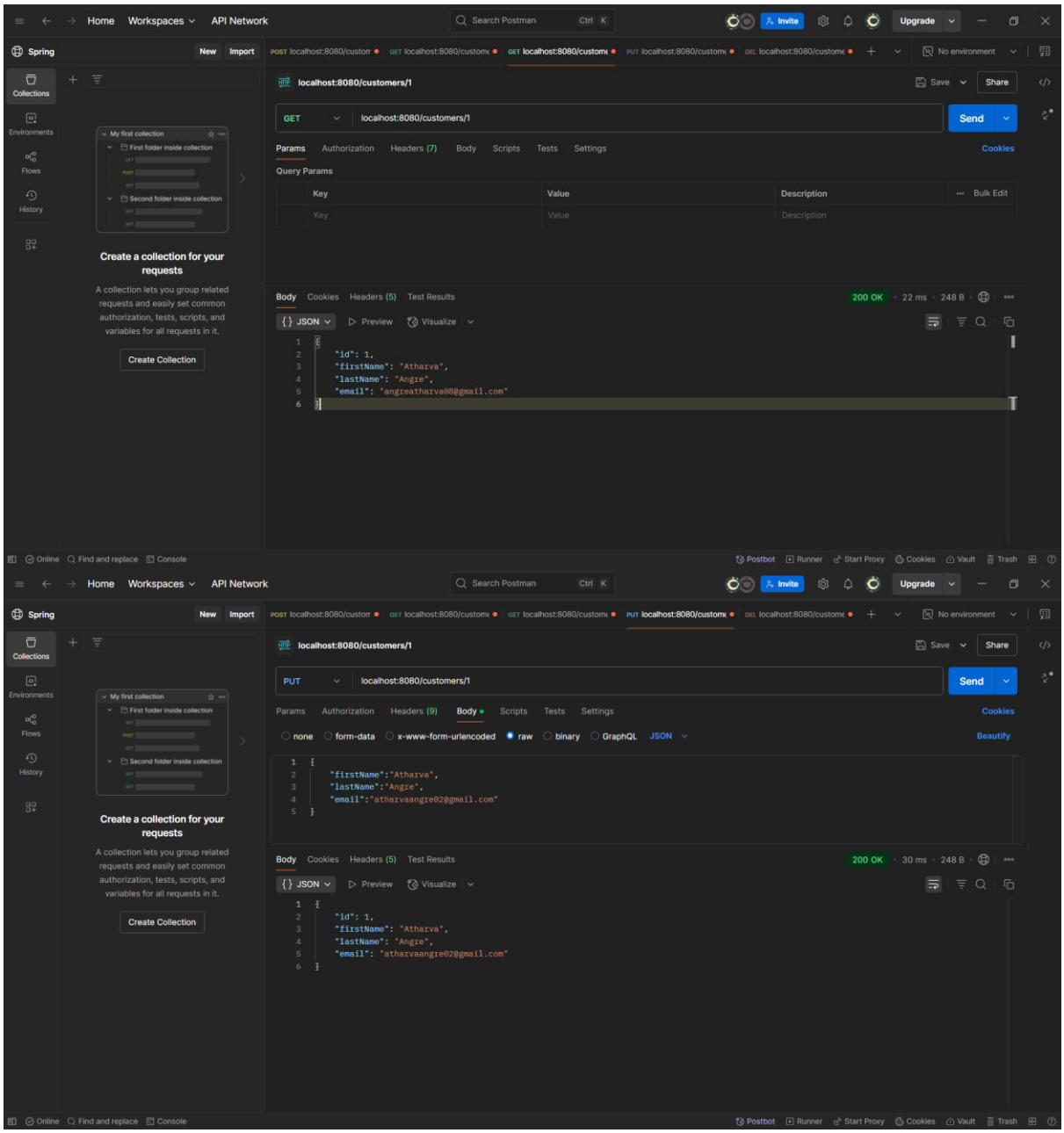
Atharva Vasant Angre **Practical 9** **2024510001**

Output:



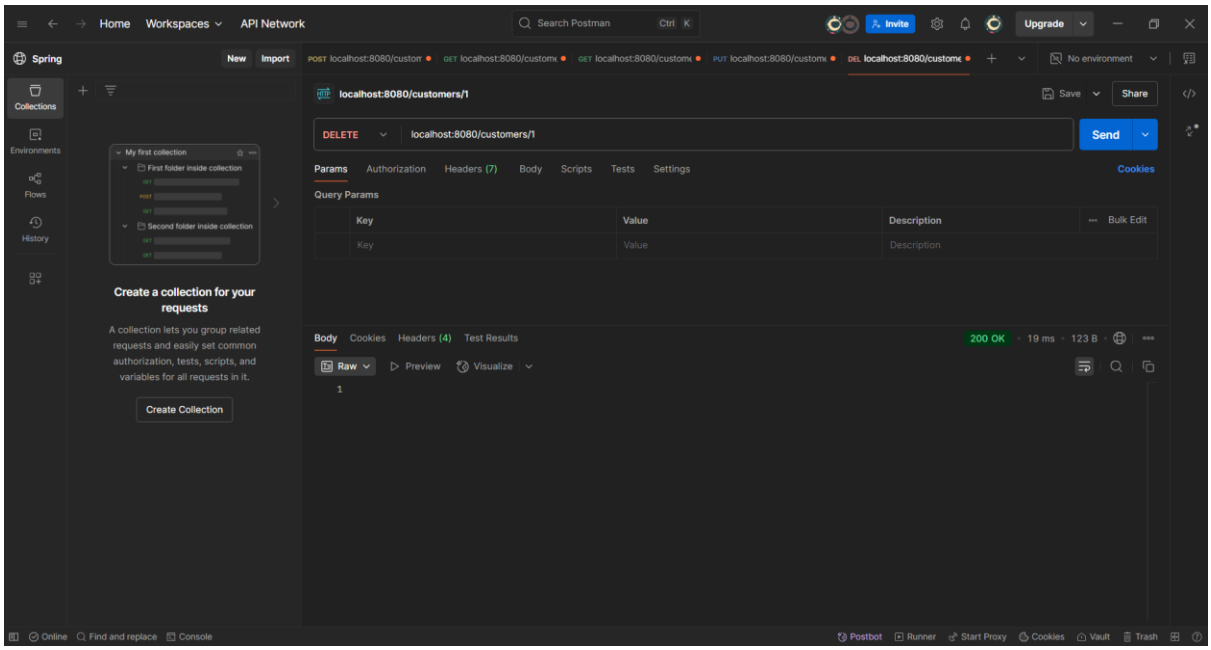
Academic Year: 2024-25 **Semester: II** **Class: FYMCA**
Course Code: MC506 **Course Name: Advanced Java Programming**

Atharva Vasant Angre **Practical 9** **2024510001**



Academic Year: 2024-25 **Semester: II** **Class: FYMCA**
Course Code: MC506 **Course Name: Advanced Java Programming**

Atharva Vasant Angre **Practical 9** **2024510001**



Observation: This expriment represents a Spring Boot-based Content Management System (CMS) that implements a RESTful API architecture. The application is built using modern Java technologies including Spring Boot 3.4.4, Spring Data JPA, and H2 database, demonstrating a robust backend infrastructure. The system follows a layered architecture with clear separation of concerns, featuring controllers for API endpoints, service layer for business logic, and data access layer for database operations. The implementation includes proper error handling through custom exceptions and follows REST API best practices. The project is well-structured with proper dependency management through Maven, making it maintainable and scalable for future enhancements.