**HANDS ON EXERCISE- WEEK3**

**Program 1: Spring Core – Setter Injection with @Component and @Autowired**

MessageService.java

import org.springframework.stereotype.Component;

@Component

public class MessageService {

public String getMessage() {

return "Setter Injection using @Autowired!";

}

}

MessagePrinter.java

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

@Component

public class MessagePrinter {

private MessageService messageService;

@Autowired

public void setMessageService(MessageService messageService) {

this.messageService = messageService;

}

public void print() {

System.out.println(messageService.getMessage());

}

}

**MainApp.java**

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.context.annotation.Configuration;

@Configuration

@ComponentScan(basePackages = "com.example")

public class MainApp {

public static void main(String[] args) {

AnnotationConfigApplicationContext ctx = new AnnotationConfigApplicationContext(MainApp.class);

MessagePrinter printer = ctx.getBean(MessagePrinter.class);

printer.print();

ctx.close();

}

}

Program 2: Spring Data JPA – Custom Finder Methods

StudentRepository.java

import org.springframework.data.jpa.repository.JpaRepository;

import java.util.List;

public interface StudentRepository extends JpaRepository<Student, Long> {

List<Student> findByName(String name);

List<Student> findByEmailContaining(String keyword);

}

StudentController.java

@GetMapping("/search")

public List<Student> findByName(@RequestParam String name) {

return service.findByName(name);

}

StudentService.java

public List<Student> findByName(String name) {

return repository.findByName(name);

}

Program 3: Spring Data JPA – OneToMany Relationship

Department.java

import jakarta.persistence.\*;

import java.util.List;

@Entity

public class Department {

@Id

@GeneratedValue

private Long id;

private String name;

@OneToMany(mappedBy = "department", cascade = CascadeType.ALL)

private List<Employee> employees;

// Getters and Setters

}

Employee.java

import jakarta.persistence.\*;

@Entity

public class Employee {

@Id

@GeneratedValue

private Long id;

private String name;

@ManyToOne

@JoinColumn(name = "department\_id")

private Department department;

// Getters and Setters

}

Repositories

public interface DepartmentRepository extends JpaRepository<Department, Long> {}

public interface EmployeeRepository extends JpaRepository<Employee, Long> {}

Program 4: Spring Data JPA – Pagination and Sorting

StudentService.java

public Page<Student> getStudents(int page, int size) {

Pageable pageable = PageRequest.of(page, size, Sort.by("name").ascending());

return repository.findAll(pageable);

}

StudentController.java

@GetMapping("/page")

public Page<Student> getPage(@RequestParam int page, @RequestParam int size) {

return service.getStudents(page, size);

}