SHRI VILEPARLE KELAVANI MANDAL'S
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC ACCREDITED with "A" GRADE (CGPA : 3.18)
DEPARTMENT OF INFORMATION TECHNOLOGY

COURSE CODE: DJS22ITL306 **DATE:5/12/2023**
COURSE NAME: Programing Laboratory 1 (Advanced Java) **CLASS: I1-Batch1**

NAME: Ayush Vinod Upadhyay
ROLL NO: I025
SAP ID: 60003220131
BRANCH: Information Technology
BATCH: 1

EXPERIMENT NO. 08

CO/LO:
CO1- Modify the behaviour of methods, classes, and interfaces at runtime.

AIM / OBJECTIVE:

Set up a spring framework and create application using the same.

PROBLEM STATEMENTS:

Setup a spring framework in any IDE, and write a program to generate result of students from given marks.

Code :

1)Student class :

src/main/java/com/example/demo/model/Student.java

```
package com.example.demo.model;

public class Student {
    private String name;
    private int marks;
    public String getResult() {
        return (marks >= 40) ? "Pass" : "Fail";
    }
}
```

COURSE CODE: DJS22ITL306 **DATE:5/12/2023**
COURSE NAME: Programing Laboratory 1 (Advanced Java) **CLASS: I1-Batch1**

2) Controlller:

src/main/java/com/example/demo/controller/ResultController.java

```java
package com.example.demo.controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;

import com.example.demo.model.Student;

@Controller
public class ResultController {

    @GetMapping("/generateResult")
    public String generateResult(Model model) {
        // Create a sample student
        Student student = new Student();
        student.setName("John Doe");
        student.setMarks(65);

        // Add student and result to the model
        model.addAttribute("student", student);
        model.addAttribute("result", student.getResult());

        // Return the Thymeleaf template name
        return "result";
    }
}
```

3) Thymeleaf Template:

src/main/resources/templates/result.html

```html
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
```

COURSE CODE: DJS22ITL306 **DATE:5/12/2023**
COURSE NAME: Programing Laboratory 1 (Advanced Java) **CLASS: I1-Batch1**

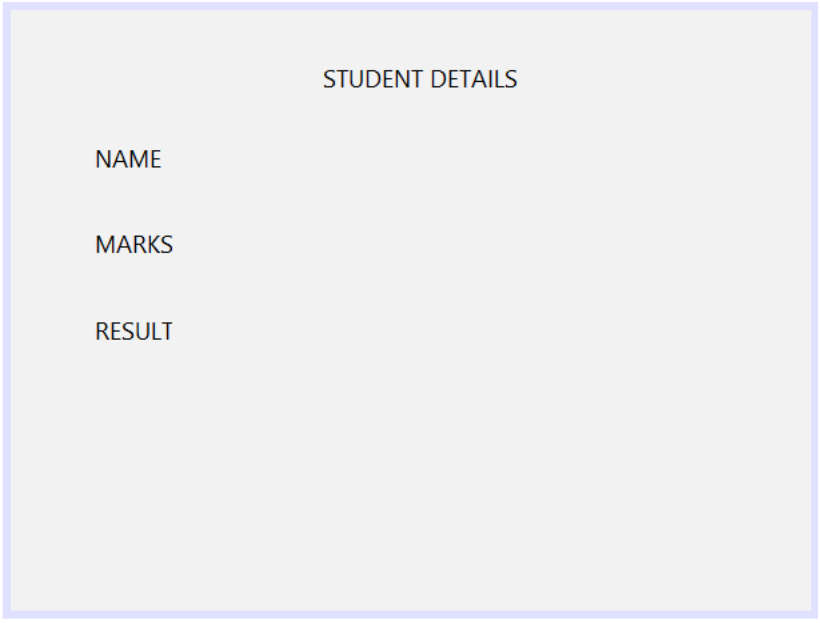
```
    <title>Student Result</title>
</head>
<body>

<h2>Student Result</h2>
<p>Name: <span th:text="${student.name}"></span></p>
<p>Marks: <span th:text="${student.marks}"></span></p>
<p>Result: <span th:text="${result}"></span></p>

</body>
</html>
```

Output:

Before execution:



After execution:

COURSE CODE: DJS22ITL306 **DATE:5/12/2023**
COURSE NAME: Programing Laboratory 1 (Advanced Java) **CLASS: I1-Batch1**

STUDENT DETAILS

NAME	Ayush Upadhyay
MARKS	65
RESULT	pass

OBSERVATION:

The Spring Framework is a comprehensive and widely used framework for Java development, offering a range of features and benefits that make it popular among developers. Here are some advantages of the Spring Framework:

Modularity:

Spring follows a modular design, allowing developers to use only the modules that are needed for their application. This modularity promotes a lightweight and flexible development approach.

Inversion of Control (IoC):

The IoC container in Spring manages the components of the application, reducing the coupling between classes. This inversion of control simplifies the integration of components and promotes a more maintainable and testable codebase.

Aspect-Oriented Programming (AOP):

Spring provides support for AOP, allowing developers to separate cross-cutting concerns such as logging, security, and transactions. AOP helps in achieving cleaner and more maintainable code by modularizing cross-cutting concerns.

Dependency Injection (DI):

Spring's DI container simplifies the process of injecting dependencies into components, making the code more modular and easier to test. It promotes loose coupling between components and enhances code readability.

COURSE CODE: DJS22ITL306 **DATE:5/12/2023**
COURSE NAME: Programing Laboratory 1 (Advanced Java) **CLASS: I1-Batch1**

Abstraction over Low-Level APIs:

Spring provides abstractions over low-level APIs like JDBC, JMS, and JTA. This abstraction simplifies the development process and promotes consistency across different technologies.

Data Access:

Spring's JDBC and ORM (Object-Relational Mapping) modules simplify data access operations. Spring provides support for various data sources, transaction management, and declarative transactions, making database interactions more efficient.

CONCLUSION:

We learned and implemented an application with spring framework of java in this experiment and understood the benefits of this framework through the observation.