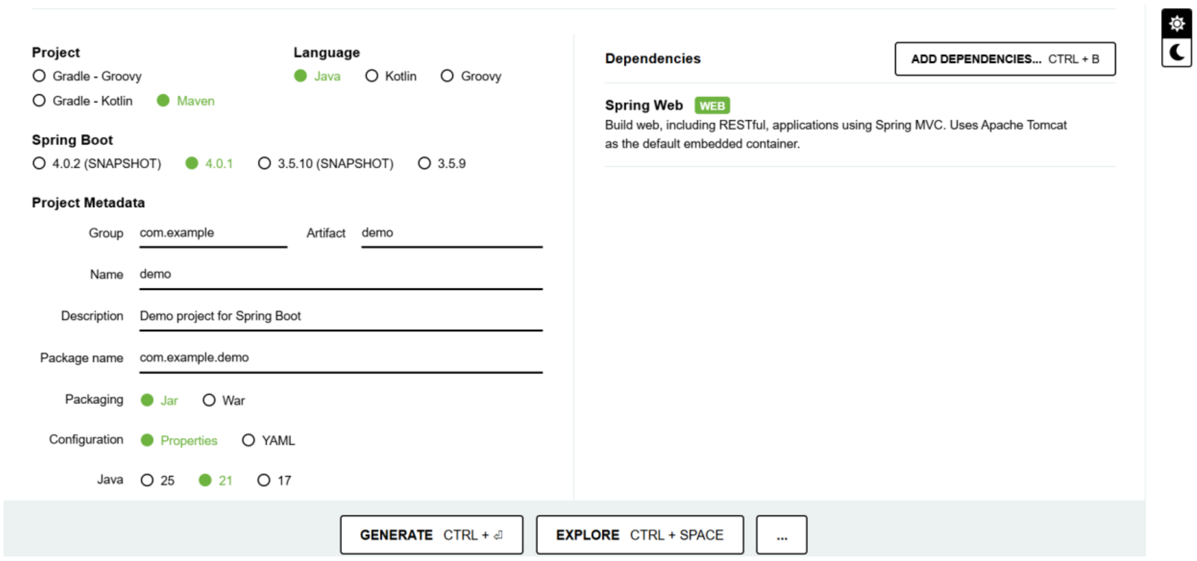
Roll no : 122

# Practical 10

## Aim: Write a program to create a simple Spring Boot application that prints a message.

**Step 1 :**

1. **Go to [https://start.spring.io/](https://start.spring.io/)**
2. **Select:**
    - **Project: Maven**
    - **Language: Java**
    - **Spring Boot: 3.x**
    - **Packaging: Jar**
    - **Java: 17**
3. **Add Dependency:**
    - **Spring Web**
4. **Click Generate**
5. **Extract ZIP file**



**Project**
Gradle - Groovy Gradle - Kotlin [✓ Maven]

**Language**
[✓ Java] Kotlin Groovy

**Spring Boot**
4.0.2 (SNAPSHOT) [✓ 4.0.1] 3.5.10 (SNAPSHOT) 3.5.9

**Project Metadata**

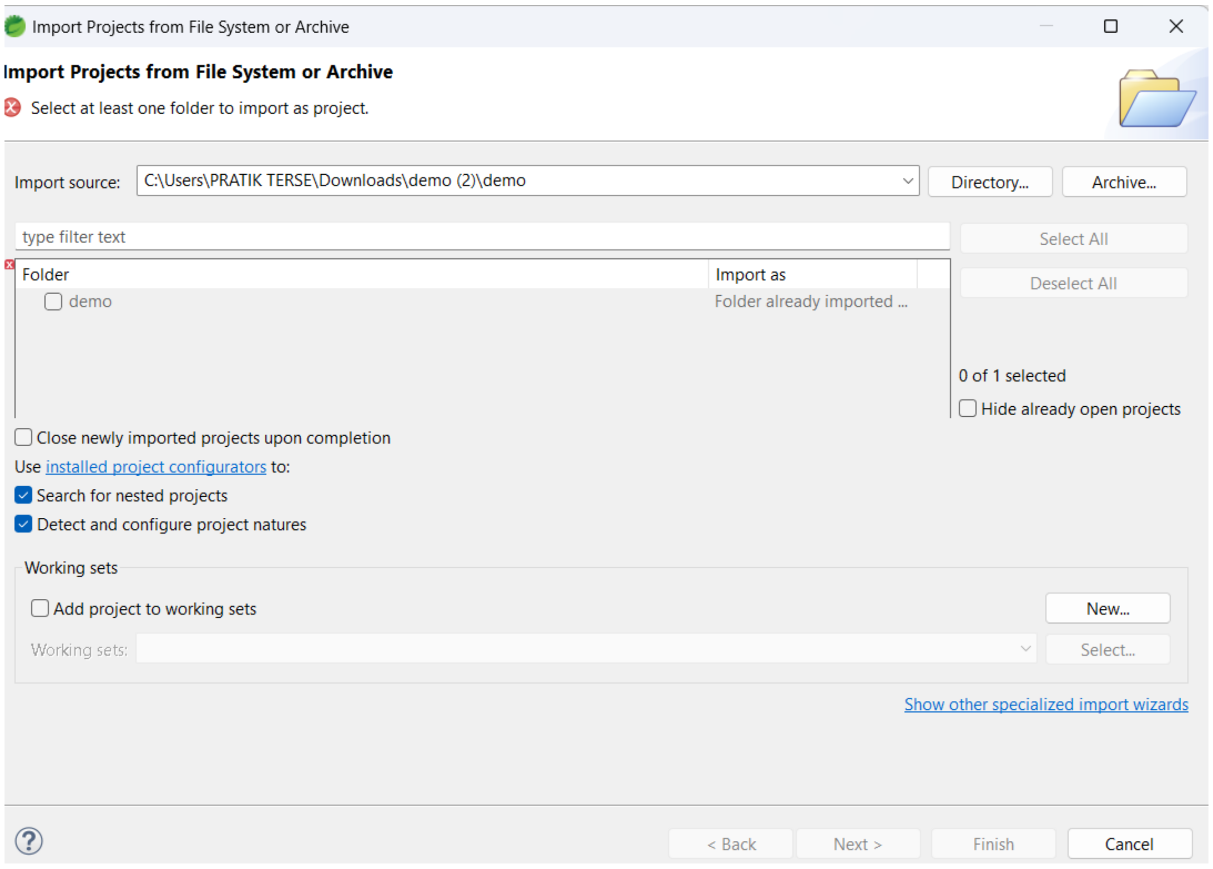
| | |
|---|---|
| Group | com.example |
| Artifact | demo |
| Name | demo |
| Description | Demo project for Spring Boot |
| Package name | com.example.demo |
| Packaging | [✓ Jar] War |
| Configuration | [✓ Properties] YAML |
| Java | 25 [✓ 21] 17 |

**Dependencies**

**Spring Web** WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

**Step 2:**

1. **Open IDE Eclipse**
2. **Right Click on File**
3. **Click on Open File From System**
4. **Click on Browse and Select Directory**
5. **Click on Finish.**



Import Projects from File System or Archive

**Import Projects from File System or Archive**

Select at least one folder to import as project.

Import source: C:\Users\PRATIK TERSE\Downloads\demo (2)\demo

type filter text

| Folder | Import as |
|---|---|
| ☐ demo | Folder already imported ... |

0 of 1 selected

☐ Hide already open projects

☐ Close newly imported projects upon completion

Use installed project configurators to:

☑ Search for nested projects

☑ Detect and configure project natures

Working sets

☐ Add project to working sets

Working sets:

**Create Java Class For Main Application + Controller**

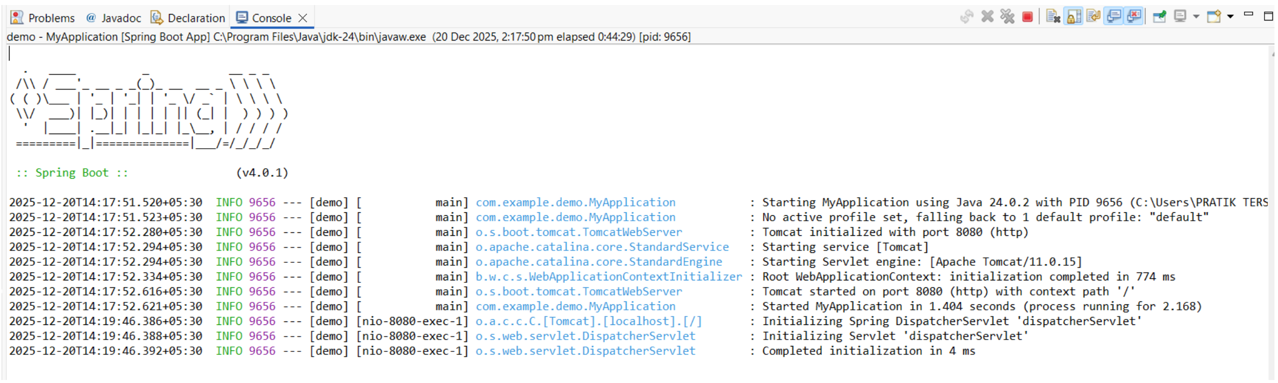
**MyApplication.java**

**Code:**

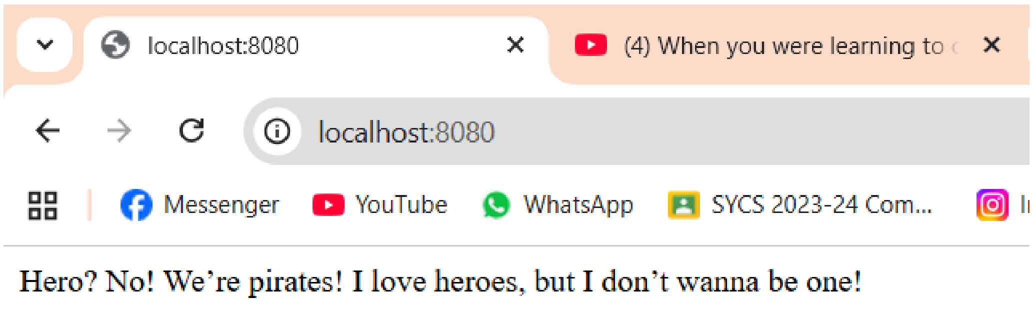
```
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@SpringBootApplication
@RestController
public class MyApplication {

   public static void main(String[] args) {
      SpringApplication.run(MyApplication.class, args);
   }

   @GetMapping("/")
   public String message() {
      return "Hero? No! We’re pirates! I love heroes, but I don’t wanna be one!";
   }
}
```

Roll no : 122

## Output:



```
demo - MyApplication [Spring Boot App] C:\Program Files\Java\jdk-24\bin\javaw.exe  (20 Dec 2025, 2:17:50 pm elapsed 0:44:29) [pid: 9656]

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/

 :: Spring Boot ::                (v4.0.1)

2025-12-20T14:17:51.520+05:30  INFO 9656 --- [demo] [           main] com.example.demo.MyApplication           : Starting MyApplication using Java 24.0.2 with PID 9656 (C:\Users\PRATIK TERS
2025-12-20T14:17:51.523+05:30  INFO 9656 --- [demo] [           main] com.example.demo.MyApplication           : No active profile set, falling back to 1 default profile: "default"
2025-12-20T14:17:52.280+05:30  INFO 9656 --- [demo] [           main] o.s.boot.tomcat.TomcatWebServer          : Tomcat initialized with port 8080 (http)
2025-12-20T14:17:52.294+05:30  INFO 9656 --- [demo] [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2025-12-20T14:17:52.294+05:30  INFO 9656 --- [demo] [           main] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [Apache Tomcat/11.0.15]
2025-12-20T14:17:52.334+05:30  INFO 9656 --- [demo] [           main] b.w.c.s.WebApplicationContextInitializer : Root WebApplicationContext: initialization completed in 774 ms
2025-12-20T14:17:52.616+05:30  INFO 9656 --- [demo] [           main] o.s.boot.tomcat.TomcatWebServer          : Tomcat started on port 8080 (http) with context path '/'
2025-12-20T14:17:52.621+05:30  INFO 9656 --- [demo] [           main] com.example.demo.MyApplication           : Started MyApplication in 1.404 seconds (process running for 2.168)
2025-12-20T14:19:46.386+05:30  INFO 9656 --- [demo] [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring DispatcherServlet 'dispatcherServlet'
2025-12-20T14:19:46.388+05:30  INFO 9656 --- [demo] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
2025-12-20T14:19:46.392+05:30  INFO 9656 --- [demo] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Completed initialization in 4 ms
```



Hero? No! We’re pirates! I love heroes, but I don’t wanna be one!

**2- Write a program to demonstrate RESTful Web Services with spring boot.**

**pom.xml**

**Code:**

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
         http://maven.apache.org/xsd/maven-4.0.0.xsd">
   <modelVersion>4.0.0</modelVersion>
   <groupId>com.example</groupId>
   <artifactId>demo</artifactId>
   <version>0.0.1-SNAPSHOT</version>
   <parent>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-parent</artifactId>
      <version>3.2.0</version>
   </parent>
   <dependencies>
      <dependency>
         <groupId>org.springframework.boot</groupId>
         <artifactId>spring-boot-starter-web</artifactId>
      </dependency>
   </dependencies>
   <build>
      <plugins>
         <plugin>
            <groupId>org.springframework.boot</groupId>
```

```
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>
</project>
```

**Main Application Class**

**DemoApplication.java**

**Code:**

```
package com.example.demo;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
@SpringBootApplication
public class DemoApplication {
    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
    }
}
```

**Model Class**

**Student.java**

**Code:**

```
package com.example.demo;
public class Student {
    private int id;
    private String name;
    private String course;
    public Student() {}
    public Student(int id, String name, String course) {
        this.id = id;
        this.name = name;
        this.course = course;
    }
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }
    public String getName() { return name; }
    public void setName(String name) { this.name = name; }
    public String getCourse() { return course; }
    public void setCourse(String course) { this.course = course; }
}
```

**REST Controller**

**StudentController.java**

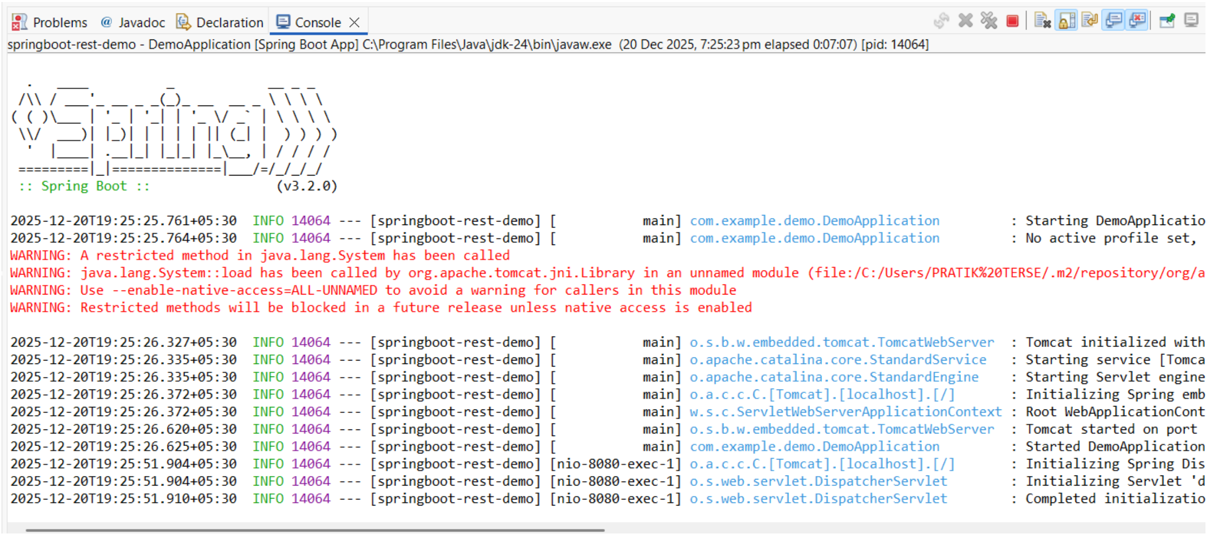
**Code:**

```
package com.example.demo;

import com.example.demo.Student;
import org.springframework.web.bind.annotation.*;

import java.util.ArrayList;
import java.util.List;
```

```
@RestController
@RequestMapping("/students")
public class StudentController {
   private List<Student> students = new ArrayList<>();
   @GetMapping
   public List<Student> getAllStudents() {
      return students;
   }

   @GetMapping("/{id}")
   public Student getStudent(@PathVariable int id) {
      return students.stream()
            .filter(s -> s.getId() == id)
            .findFirst()
            .orElse(null);
   }
   @PostMapping
   public String addStudent(@RequestBody Student student) {
      students.add(student);
      return "Student Added Successfully";
   }
   @PutMapping("/{id}")
   public String updateStudent(@PathVariable int id, @RequestBody Student
student) {
      for (Student s : students) {
         if (s.getId() == id) {
            s.setName(student.getName());
```

```
                s.setCourse(student.getCourse());
                return "Student Updated";
            }
        }
        return "Student Not Found";
    }
    @DeleteMapping("/{id}")
    public String deleteStudent(@PathVariable int id) {
        students.removeIf(s -> s.getId() == id);
        return "Student Deleted";
    }
}
```

**Output:**

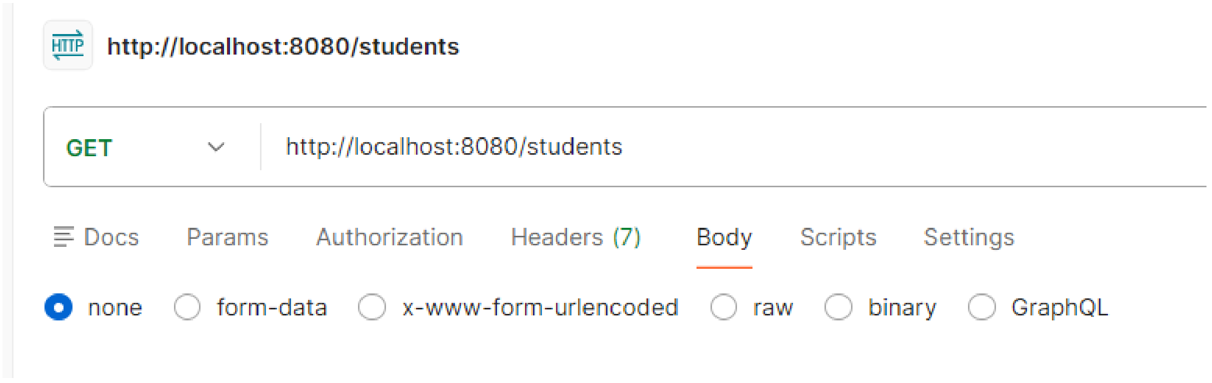


```
springboot-rest-demo - DemoApplication [Spring Boot App] C:\Program Files\Java\jdk-24\bin\javaw.exe  (20 Dec 2025, 7:25:23 pm elapsed 0:07:07) [pid: 14064]

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v3.2.0)

2025-12-20T19:25:25.761+05:30  INFO 14064 --- [springboot-rest-demo] [           main] com.example.demo.DemoApplication         : Starting DemoApplicatio
2025-12-20T19:25:25.764+05:30  INFO 14064 --- [springboot-rest-demo] [           main] com.example.demo.DemoApplication         : No active profile set,
WARNING: A restricted method in java.lang.System has been called
WARNING: java.lang.System::load has been called by org.apache.tomcat.jni.Library in an unnamed module (file:/C:/Users/PRATIK%20TERSE/.m2/repository/org/a
WARNING: Use --enable-native-access=ALL-UNNAMED to avoid a warning for callers in this module
WARNING: Restricted methods will be blocked in a future release unless native access is enabled

2025-12-20T19:25:26.327+05:30  INFO 14064 --- [springboot-rest-demo] [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with
2025-12-20T19:25:26.335+05:30  INFO 14064 --- [springboot-rest-demo] [           main] o.apache.catalina.core.StandardService   : Starting service [Tomca
2025-12-20T19:25:26.335+05:30  INFO 14064 --- [springboot-rest-demo] [           main] o.apache.catalina.core.StandardEngine    : Starting Servlet engine
2025-12-20T19:25:26.372+05:30  INFO 14064 --- [springboot-rest-demo] [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring emb
2025-12-20T19:25:26.372+05:30  INFO 14064 --- [springboot-rest-demo] [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationCont
2025-12-20T19:25:26.620+05:30  INFO 14064 --- [springboot-rest-demo] [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port
2025-12-20T19:25:26.625+05:30  INFO 14064 --- [springboot-rest-demo] [           main] com.example.demo.DemoApplication         : Started DemoApplication
2025-12-20T19:25:51.904+05:30  INFO 14064 --- [springboot-rest-demo] [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring Dis
2025-12-20T19:25:51.904+05:30  INFO 14064 --- [springboot-rest-demo] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'd
2025-12-20T19:25:51.910+05:30  INFO 14064 --- [springboot-rest-demo] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet        : Completed initializatio
```

**Roll no : 122**

**Postman:**

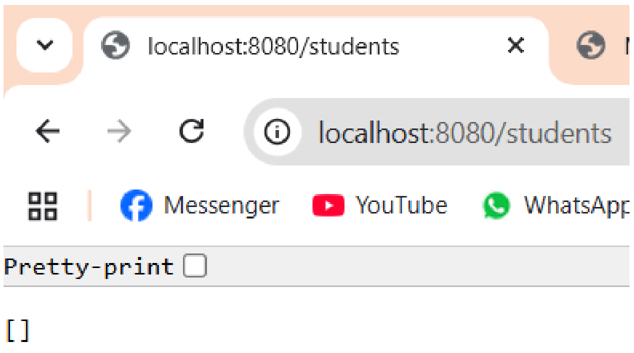
**Get Method:**



http://localhost:8080/students

GET http://localhost:8080/students

Body: [✓ none] form-data x-www-form-urlencoded raw binary GraphQL

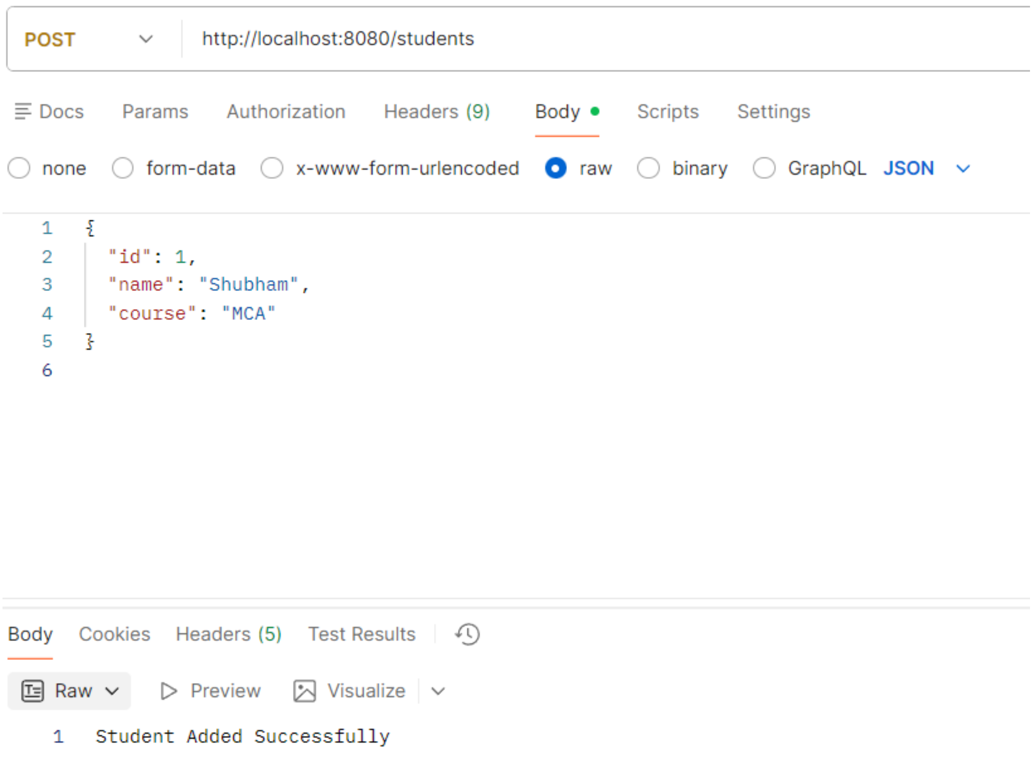


localhost:8080/students

```
[]
```

Roll no : 122

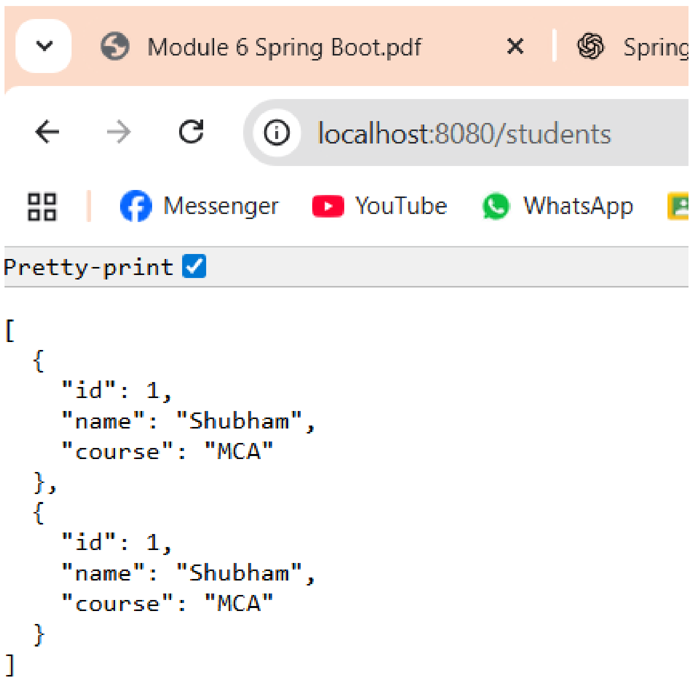
## Post Method:



http://localhost:8080/students

```
{
  "id": 1,
  "name": "Shubham",
  "course": "MCA"
}
```

```
Student Added Successfully
```



localhost:8080/students

```
[
  {
    "id": 1,
    "name": "Shubham",
    "course": "MCA"
  },
  {
    "id": 1,
    "name": "Shubham",
    "course": "MCA"
  }
]
```

**3. Write a program to demonstrate Database Connection with spring.**

```
CREATE DATABASE testdb;
USE testdb;

CREATE TABLE user (
   id INT PRIMARY KEY,
   name VARCHAR(255),
   email VARCHAR(255)
);
```

**Pom.xml**

**Code:**

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
       https://maven.apache.org/xsd/maven-4.0.0.xsd">
   <modelVersion>4.0.0</modelVersion>
   <!-- Spring Boot Parent -->
   <parent>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-parent</artifactId>
      <version>3.2.5</version>
      <relativePath/>
   </parent>
   <groupId>com.example</groupId>
   <artifactId>demo</artifactId>
```

```
<version>0.0.1-SNAPSHOT</version>
<name>demo</name>
<description>Spring Boot Demo Project</description>
<properties>
  <java.version>17</java.version>
</properties>
<dependencies>
  <!-- Spring Web -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>
  <!-- Spring Data JPA -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
  </dependency>
  <!-- MySQL Driver -->
  <dependency>
    <groupId>com.mysql</groupId>
    <artifactId>mysql-connector-j</artifactId>
    <scope>runtime</scope>
  </dependency>
  <!-- Testing -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
```

```
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>
    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>
</project>
```

**src/main/resources - > application.properties**

**Code:**

```
spring.datasource.url=jdbc:mysql://localhost:3306/testdb
spring.datasource.username=root
spring.datasource.password=Shubham@29
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.database-platform=org.hibernate.dialect.MySQL8Dialect
server.port=8080
```

**Main Class – DemoApplication.java**

**Code:**

```
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DemoApplication {

  public static void main(String[] args) {
    SpringApplication.run(DemoApplication.class, args);
    System.out.println("Application Started Successfully!");
  }
}
```

**Entity Class – User.java**

**Code:**

```
package com.example.demo;
import jakarta.persistence.Entity;
import jakarta.persistence.Id;
@Entity
public class User {
  @Id
  private int id;
  private String name;
  private String email;
  public int getId() { return id; }
```

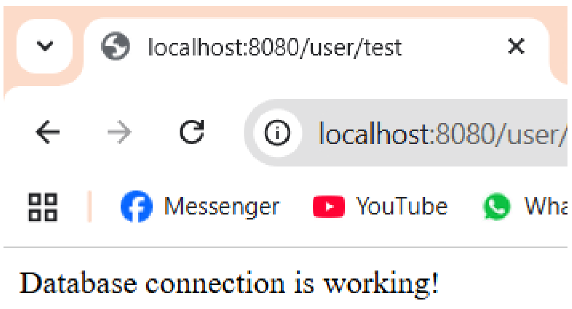
```
    public void setId(int id) { this.id = id; }
    public String getName() { return name; }
    public void setName(String name) { this.name = name; }
    public String getEmail() { return email; }
    public void setEmail(String email) { this.email = email; }
}
```

**Repository Interface – UserRepository.java**

**Code:**

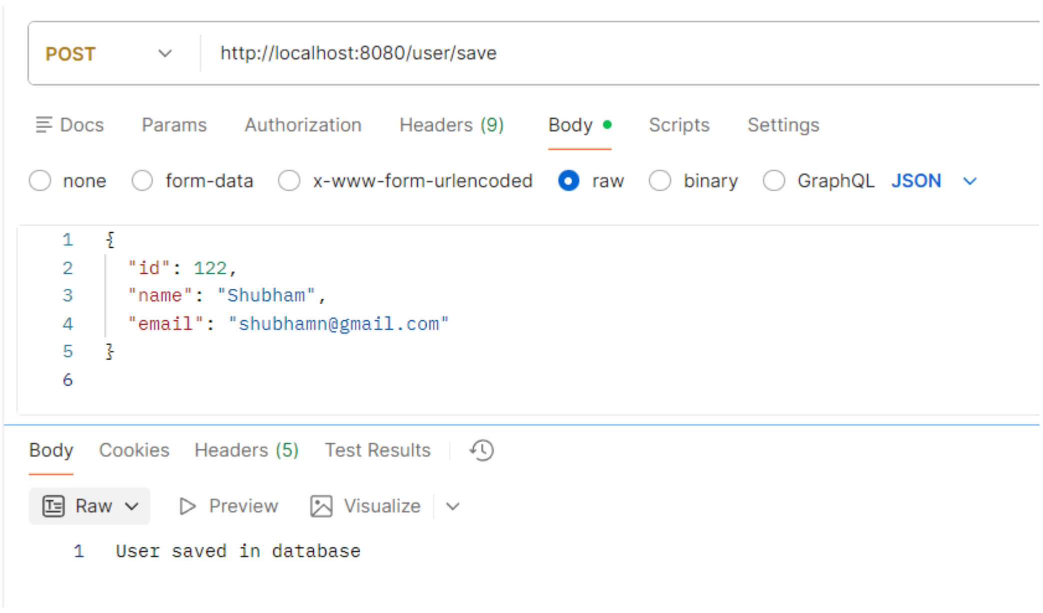
```
package com.example.demo;
import org.springframework.data.jpa.repository.JpaRepository;
public interface UserRepository extends JpaRepository<User, Integer> {
}
```

**Output:**



```
Database connection is working!
```

**Roll no : 122**

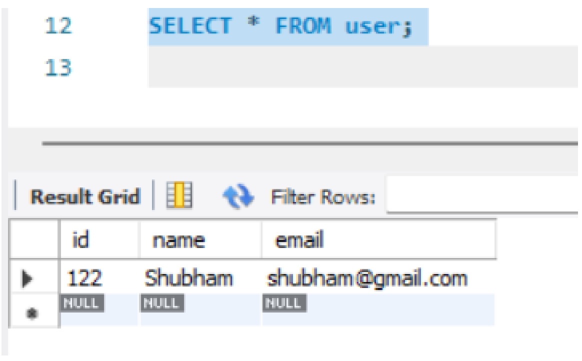


POST http://localhost:8080/user/save

Body: raw [✓] JSON

```
{
  "id": 122,
  "name": "Shubham",
  "email": "shubhamn@gmail.com"
}
```

```
User saved in database
```



```
SELECT * FROM user;
```

| id | name | email |
|---|---|---|
| 122 | Shubham | shubham@gmail.com |
| NULL | NULL | NULL |