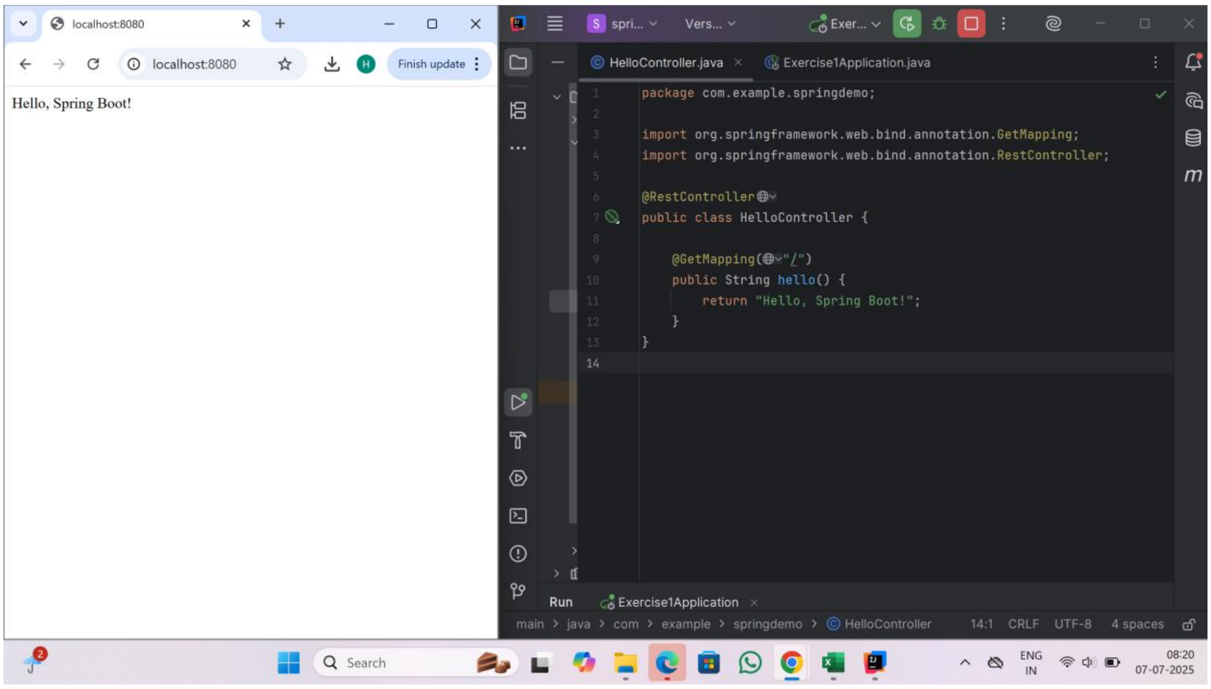
Exercise 1: Configuring a Basic Spring Application

Code:

```
package com.example.springdemo;

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

@RestController
public class HelloController {

    @GetMapping("/")
    public String hello() {
        return "Hello, Spring Boot!";
    }
}
```

Output:


localhost:8080
HelloController.java
Exercise1Application.java
Run
Exercise1Application
main > java > com > example > springdemo > HelloController

```
Hello, Spring Boot!
```

```
package com.example.springdemo;

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

@RestController
public class HelloController {

    @GetMapping("/")
    public String hello() {
        return "Hello, Spring Boot!";
    }
}
```

Exercise 2: Implementing Dependency Injection

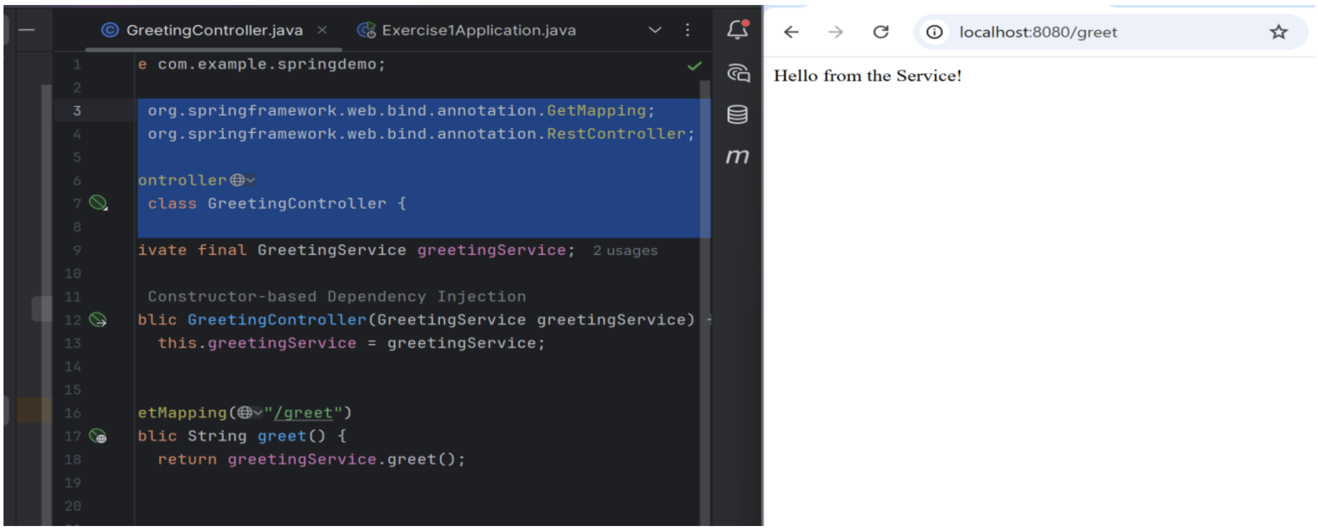
ServiceClass Code:

```
package com.example.springdemo;

import org.springframework.stereotype.Service;

@Service

public class GreetingService {

    public String greet() {

        return "Hello from the Service!";

    }

}
```

ControllerClass Code:

```
package com.example.springdemo;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;
@RestController
public class GreetingController {

    private final GreetingService greetingService;
    public GreetingController(GreetingService greetingService) {
        this.greetingService = greetingService;
    }
    @GetMapping("/greet")
    public String greet() {
        return greetingService.greet();
    }
}
```

Output:



```
e com.example.springdemo;

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

ontroller
 class GreetingController {

ivate final GreetingService greetingService;  2 usages

 Constructor-based Dependency Injection
blic GreetingController(GreetingService greetingService)
  this.greetingService = greetingService;


etMapping("/greet")
blic String greet() {
  return greetingService.greet();
```

localhost:8080/greet

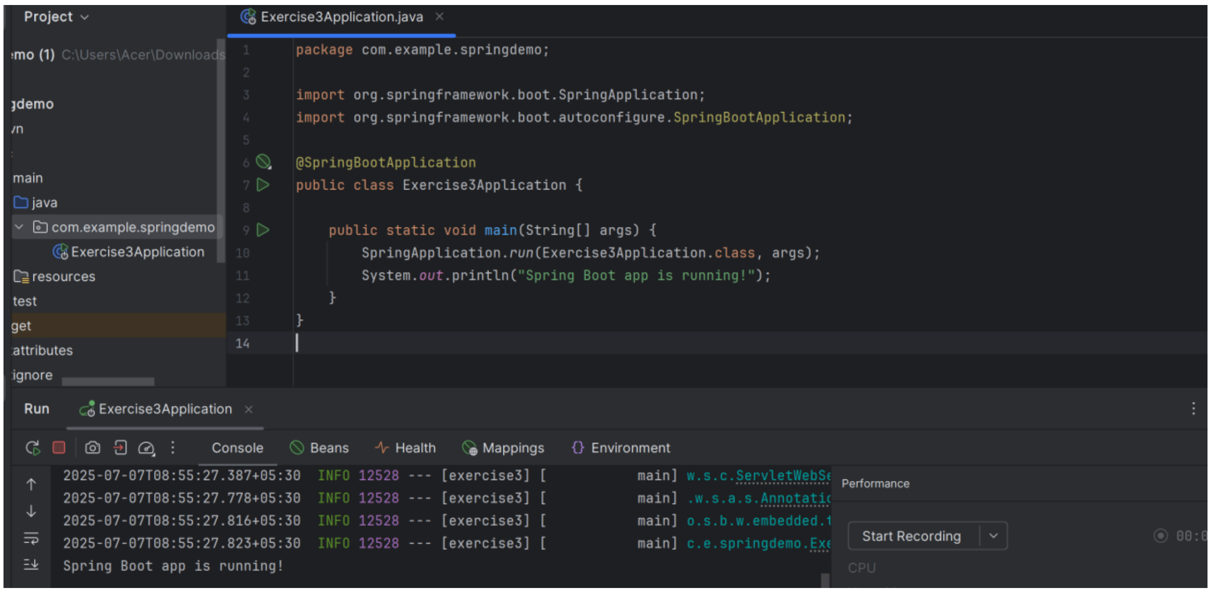
Hello from the Service!

Exercise 3: Creating and Configuring a Maven Project

Code:

```
package com.example.springdemo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class Exercise3Application {

    public static void main(String[] args) {

        SpringApplication.run(Exercise3Application.class, args);

        System.out.println("Spring Boot app is running!");

    }

}
```

Output:



```java
package com.example.springdemo;

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

@SpringBootApplication
public class Exercise3Application {

    public static void main(String[] args) {
        SpringApplication.run(Exercise3Application.class, args);
        System.out.println("Spring Boot app is running!");
    }
}

```

```
2025-07-07T08:55:27.387+05:30  INFO 12528 --- [exercise3] [           main] w.s.c.ServletWebSe
2025-07-07T08:55:27.778+05:30  INFO 12528 --- [exercise3] [           main] .w.s.a.s.Annotatic
2025-07-07T08:55:27.816+05:30  INFO 12528 --- [exercise3] [           main] o.s.b.w.embedded.t
2025-07-07T08:55:27.823+05:30  INFO 12528 --- [exercise3] [           main] c.e.springdemo.Exe
Spring Boot app is running!
```

Exercise 4: Spring Data JPA - Quick Example

Dependency Check:

```xml
<?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>
  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>3.5.3</version>
    <relativePath/> <!-- lookup parent from repository -->
  </parent>
  <groupId>com.example</groupId>
  <artifactId>springdemo</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>exercise3</name>
  <description>Demo project for Spring Boot</description>
  <url/>
```

```xml
<licenses>
  <license/>
</licenses>
<developers>
  <developer/>
</developers>
<scm>
  <connection/>
  <developerConnection/>
  <tag/>
  <url/>
</scm>
<properties>
  <java.version>17</java.version>
</properties>
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web-services</artifactId>
  </dependency>

  <dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <scope>runtime</scope>
  </dependency>
  <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>
```

StudentClass:

```java
package com.example.springdemo;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;

@Entity
public class Student {

   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Long id;

   private String name;
   private String email;

   public Student() {
   }

   public Student(String name, String email) {
      this.name = name;
      this.email = email;
   }

   // Getters & Setters

   public Long getId() {
      return id;
   }

   public String getName() {
      return name;
```

```
    }

    public String getEmail() {
        return email;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void setEmail(String email) {
        this.email = email;
    }
}
```

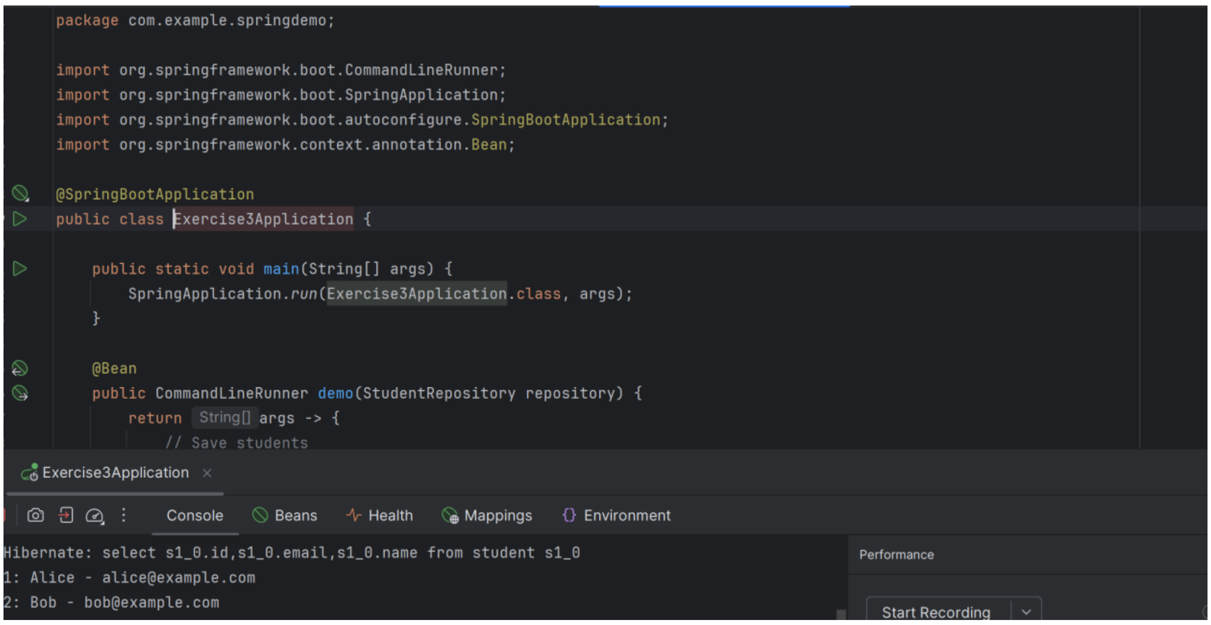
StudentRepository Class:

```
package com.example.springdemo;

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

public interface StudentRepository extends JpaRepository<Student, Long> {
}
```

Main Class:

```
package com.example.springdemo;

import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean;

@SpringBootApplication
public class Exercise3Application {

    public static void main(String[] args) {
       SpringApplication.run(Exercise3Application.class, args);
    }
    @Bean
    public CommandLineRunner demo(StudentRepository repository) {
      return args -> {
        // Save students
        repository.save(new Student("Alice", "alice@example.com"));
        repository.save(new Student("Bob", "bob@example.com"));

        // Fetch and print
        System.out.println("Students in DB:");
        for (Student student : repository.findAll()) {
          System.out.println(student.getId() + ": " + student.getName() + " - " +
student.getEmail());
        }
      };
    }
}
```

Output:



```
package com.example.springdemo;

import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean;

@SpringBootApplication
public class Exercise3Application {

    public static void main(String[] args) {
        SpringApplication.run(Exercise3Application.class, args);
    }

    @Bean
    public CommandLineRunner demo(StudentRepository repository) {
        return String[] args -> {
            // Save students
```

```
Hibernate: select s1_0.id,s1_0.email,s1_0.name from student s1_0
1: Alice - alice@example.com
2: Bob - bob@example.com
```