

SPRING REST USING SPRING BOOT 3

Create a Spring Web Project using Maven

spring-learn\spring-learn\src\main\java\com\cognizant\spring_learn\SpringLearnApplication.java

```
package com.cognizant.spring_learn;
```

```
import org.springframework.boot.SpringApplication;
```

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

```
@SpringBootApplication
```

```
public class SpringLearnApplication {
```

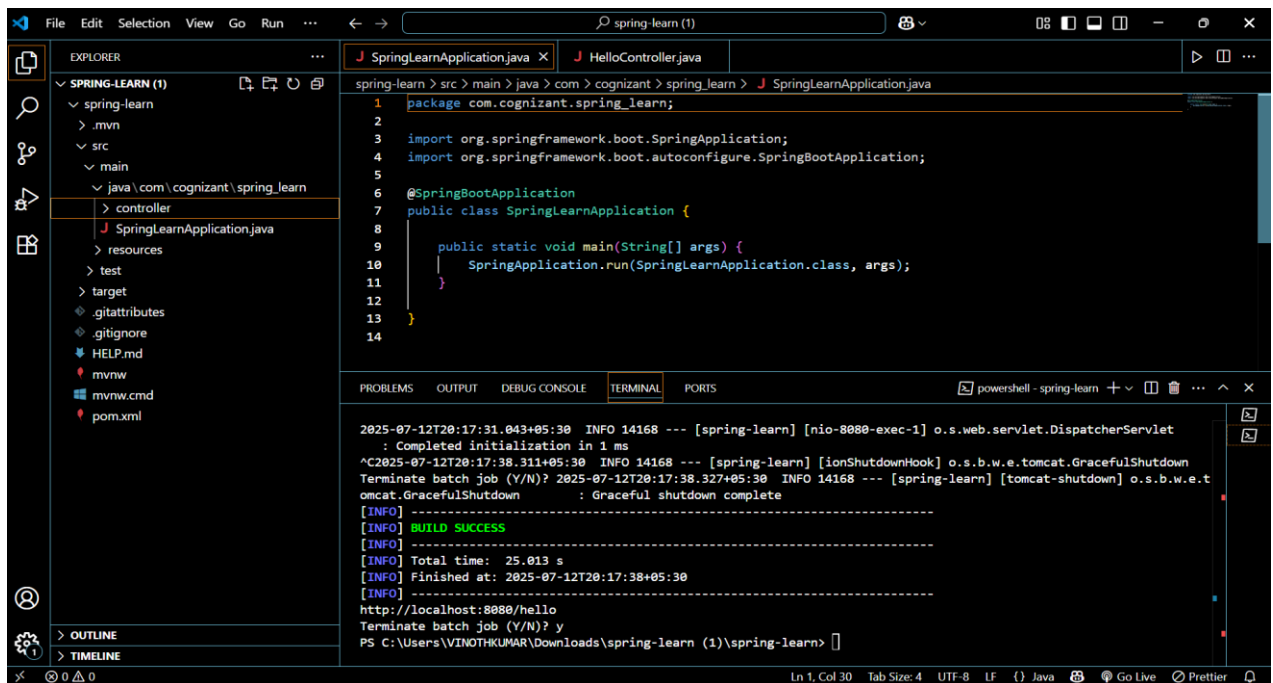
```
    public static void main(String[] args) {
```

```
        SpringApplication.run(SpringLearnApplication.class, args);
```

```
    }
```

```
}
```

OUTPUT :



The screenshot shows an IDE with the following components:

- EXPLORER:** Displays the project structure for 'spring-learn (1)', including folders like .mvn, src, main, controller, resources, test, target, .gitattributes, .gitignore, HELP.md, mvnw, mvnw.cmd, and pom.xml.
- SpringLearnApplication.java:** The source code file is open, showing the package declaration, imports, and the main method.
- TERMINAL:** Displays the output of the application run, including logs for initialization, shutdown, and successful completion.

```
1 package com.cognizant.spring_learn;
2
3 import org.springframework.boot.SpringApplication;
4 import org.springframework.boot.autoconfigure.SpringBootApplication;
5
6 @SpringBootApplication
7 public class SpringLearnApplication {
8
9     public static void main(String[] args) {
10         SpringApplication.run(SpringLearnApplication.class, args);
11     }
12
13 }
14
```

Terminal Output:

```
2025-07-12T20:17:31.043+05:30 INFO 14168 --- [spring-learn] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet
: Completed initialization in 1 ms
^C2025-07-12T20:17:38.311+05:30 INFO 14168 --- [spring-learn] [ionShutdownHook] o.s.b.w.e.tomcat.GracefulShutdown
Terminate batch job (Y/N)? 2025-07-12T20:17:38.327+05:30 INFO 14168 --- [spring-learn] [tomcat-shutdown] o.s.b.w.e.t
omcat.GracefulShutdown : Graceful shutdown complete
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 25.013 s
[INFO] Finished at: 2025-07-12T20:17:38+05:30
[INFO] -----
http://localhost:8080/hello
Terminate batch job (Y/N)? y
PS C:\Users\VINOTHKUMAR\Downloads\spring-learn (1)\spring-learn>
```

Spring Core – Load Country from Spring Configuration XML

spring-learn\spring-

learn\src\main\java\com\cognizant\spring_learn\SpringLearnApplication.java

```
package com.cognizant.spring_learn;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

@SpringBootApplication
public class SpringLearnApplication {

    private static final Logger LOGGER =
        LoggerFactory.getLogger(SpringLearnApplication.class);

    public static void main(String[] args) {

        SpringApplication.run(SpringLearnApplication.class, args);

        displayCountry();

    }

    public static void displayCountry() {

        ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");

        Country country = context.getBean("country", Country.class);

        LOGGER.debug("Country : {}", country.toString());

    }

}
```

country.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.springframework.org/schema/beans
```

<https://www.springframework.org/schema/beans/spring-beans.xsd>">

```
<bean id="country" class="com.cognizant.spring_learn.Country">
```

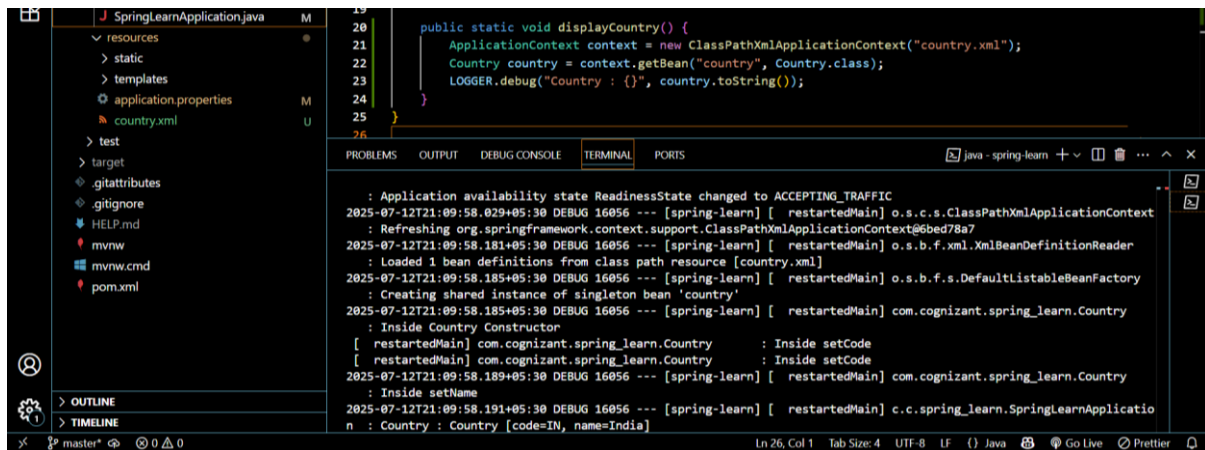
```
<property name="code" value="IN"/>
```

```
<property name="name" value="India"/>
```

```
</bean>
```

```
</beans>
```

OUTPUT :



```
public static void displayCountry() {
    ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");
    Country country = context.getBean("country", Country.class);
    LOGGER.debug("Country : {}", country.toString());
}
```

```
: Application availability state ReadinessState changed to ACCEPTING_TRAFFIC
2025-07-12T21:09:58.029+05:30 DEBUG 16056 --- [spring-learn] [ restartedMain] o.s.c.s.ClassPathXmlApplicationContext
: Refreshing org.springframework.context.support.ClassPathXmlApplicationContext@6bed78a7
2025-07-12T21:09:58.181+05:30 DEBUG 16056 --- [spring-learn] [ restartedMain] o.s.b.f.xml.XmlBeanDefinitionReader
: Loaded 1 bean definitions from class path resource [country.xml]
2025-07-12T21:09:58.185+05:30 DEBUG 16056 --- [spring-learn] [ restartedMain] o.s.b.f.s.DefaultListableBeanFactory
: Creating shared instance of singleton bean 'country'
2025-07-12T21:09:58.185+05:30 DEBUG 16056 --- [spring-learn] [ restartedMain] com.cognizant.spring_learn.Country
: Inside Country Constructor
[ restartedMain] com.cognizant.spring_learn.Country : Inside setCode
[ restartedMain] com.cognizant.spring_learn.Country : Inside setCode
2025-07-12T21:09:58.189+05:30 DEBUG 16056 --- [spring-learn] [ restartedMain] com.cognizant.spring_learn.Country
: Inside setName
2025-07-12T21:09:58.191+05:30 DEBUG 16056 --- [spring-learn] [ restartedMain] c.c.spring_learn.SpringLearnApplicatio
n : Country : Country [code=IN, name=India]
```

Hello World RESTful Web Service

spring-learn\spring-learn\src\main\java\com\cognizant\spring_learn\controller\HelloController.java

```
package com.cognizant.spring_learn.controller;
```

```
import org.slf4j.Logger;
```

```
import org.slf4j.LoggerFactory;
```

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

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

```
@RestController
```

```
public class HelloController {
```

```
    private static final Logger LOGGER = LoggerFactory.getLogger(HelloController.class);
```

```
    @GetMapping("/hello")
```

```

public String sayHello() {

    LOGGER.info("START - sayHello()");

    String message = "Hello World!!";

    LOGGER.info("END - sayHello()");

    return message;

}
}

```

OUTPUT :

The screenshot shows an IDE with the following components:

- EXPLORER:** Displays the project structure for 'spring-learn (1)', including 'src/main/java/com/cognizant/spring_learn/controller' and 'resources'.
- EDITOR:** Shows the code for 'HelloController.java' with the following content:


```

9 public class HelloController {
10
11     private static final Logger LOGGER = LoggerFactory.getLogger(HelloController.class);
12
13     @GetMapping("/hello")
14     public String sayHello() {
15         LOGGER.info("START - sayHello()");
16         String message = "Hello World!!";
17         LOGGER.info("END - sayHello()");
18         return message;
19     }
20 }
21

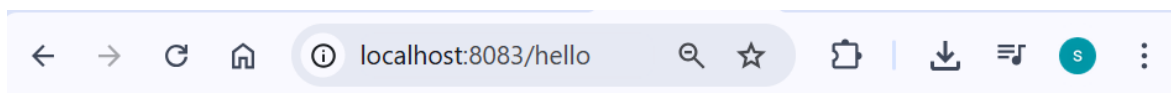
```
- TERMINAL:** Displays the output of the application, including a warning about port 8083 and build logs:


```

Web server failed to start. Port 8083 was already in use.
Action:
Identify and stop the process that's listening on port 8083 or configure this application to listen on another port.

[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 5.258 s
[INFO] Finished at: 2025-07-12T21:28:16+05:30
[INFO] -----
PS C:\Users\VINOTHKUMAR\Downloads\spring-learn (1)\spring-learn>

```



Hello World!!

REST - Country Web Service

spring-learn\spring-learn\src\main\java\com\cognizant\spring_learn\controller\CountryController.java

```
package com.cognizant.spring_learn.controller;
```

```
import com.cognizant.spring_learn.Country;
```

```

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

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

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

@RestController

public class CountryController {

    private static final Logger LOGGER = LoggerFactory.getLogger(CountryController.class);

    @RequestMapping("/country")

    public Country getCountryIndia() {

        LOGGER.info("START - getCountryIndia()");

        ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");

        Country country = context.getBean("country", Country.class);

        LOGGER.info("END - getCountryIndia()");

        return country;

    }

}

```

spring-learn\spring-learn\src\main\java\com\cognizant\spring_learn\Country.java

```

package com.cognizant.spring_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class Country {

    private static final Logger LOGGER = LoggerFactory.getLogger(Country.class);

    private String code;

    private String name;

    public Country() {

```

```
        LOGGER.debug("Inside Country Constructor");
    }

    public String getCode() {
        LOGGER.debug("Inside getCode");
        return code;
    }

    public void setCode(String code) {
        LOGGER.debug("Inside setCode");
        this.code = code;
    }

    public String getName() {
        LOGGER.debug("Inside getName");
        return name;
    }

    public void setName(String name) {
        LOGGER.debug("Inside setName");
        this.name = name;
    }

    @Override
    public String toString() {
        return "Country [code=" + code + ", name=" + name + "]";
    }
}
```

OUTPUT :

The screenshot shows an IDE with the following components:

- EXPLORER:** Displays the project structure for 'spring-learn (1)'. The file 'Country.java' is selected under the 'controller' directory.
- Source Code:** The code for 'Country.java' is visible, showing a 'Country' class with a 'setName' method and an overridden 'toString' method.
- TERMINAL:** Shows the output of a command. It indicates that the web server failed to start because port 8083 was already in use. It provides instructions to identify and stop the process listening on that port or configure the application to listen on a different port. The terminal also shows build success messages and the current directory path.

```
public class Country {
    public void setName(String name) {
        // ...
    }
    @Override
    public String toString() {
        return "Country [code=" + code + ", name=" + name + "];";
    }
}
```

Description:

Web server failed to start. Port 8083 was already in use.

Action:

Identify and stop the process that's listening on port 8083 or configure this application to listen on another port.

```
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 5.062 s
[INFO] Finished at: 2025-07-12T21:28:24+05:30
[INFO] -----
PS C:\Users\VINOTHKUMAR\Downloads\spring-learn (1)\spring-learn>
```

The screenshot shows a web client interface with the following components:

- Overview:** Displays the current request as a GET to `http://localhost:8083/country`.
- Params:** A table for query parameters.
- Body:** Shows the response body as JSON, with a status of 200 OK, 50 ms, and 192 B.
- Response Data:** A table showing the JSON response structure.

Key	Value	Desc...	Bulk Edit
Key	Value	Description	

code	IN
name	India

REST - Get country based on country code

spring-learn\spring-

learn\src\main\java\com\cognizant\spring_learn\SpringLearnApplication.java

```
package com.cognizant.spring_learn;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

@SpringBootApplication
public class SpringLearnApplication {

    private static final Logger LOGGER =
        LoggerFactory.getLogger(SpringLearnApplication.class);

    public static void main(String[] args) {

        SpringApplication.run(SpringLearnApplication.class, args);

        displayCountry();

    }

    public static void displayCountry() {

        ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");

        Country country = context.getBean("country", Country.class);

        LOGGER.debug("Country : {}", country.toString());

    }

}
```

spring-learn\spring-

learn\src\main\java\com\cognizant\spring_learn\service\CountryService.java

```
package com.cognizant.spring_learn.service;

import com.cognizant.spring_learn.Country;
import org.springframework.context.ApplicationContext;
```



```
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.stereotype.Service;
import java.util.List;

@Service

public class CountryService {

    public Country getCountry(String code) {

        ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");

        List<Country> countryList = (List<Country>) context.getBean("countryList");

        return countryList.stream()

            .filter(c -> c.getCode().equalsIgnoreCase(code))

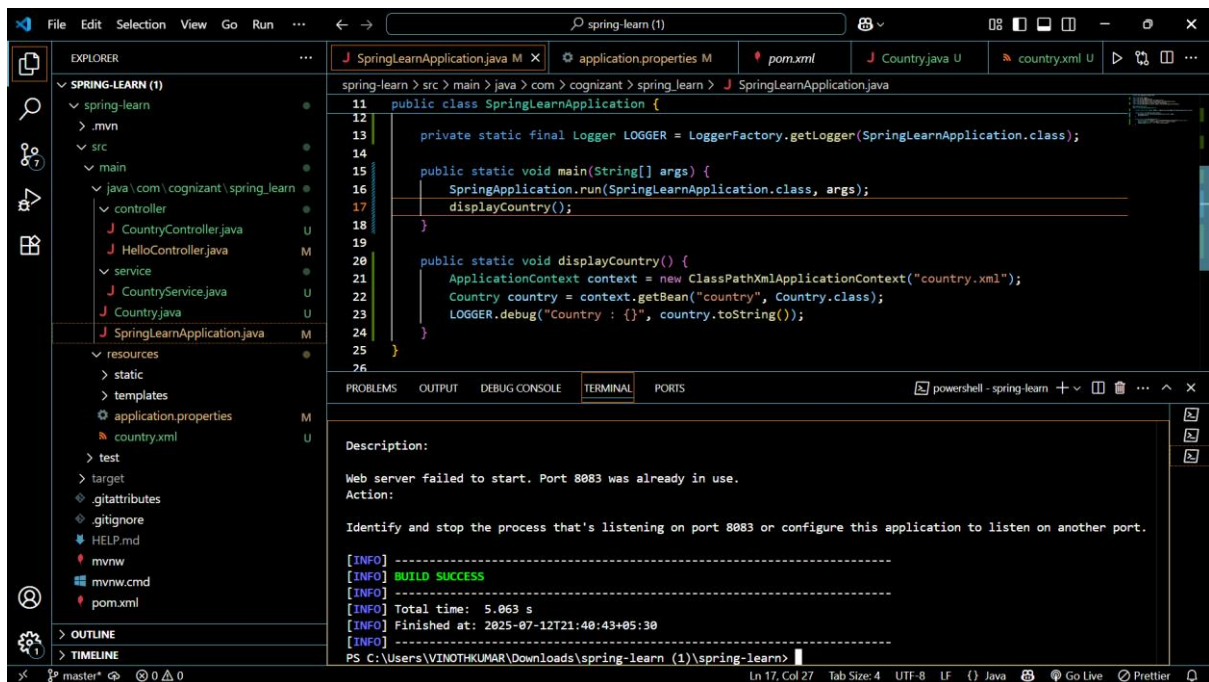
            .findFirst()

            .orElse(null);

    }

}
```

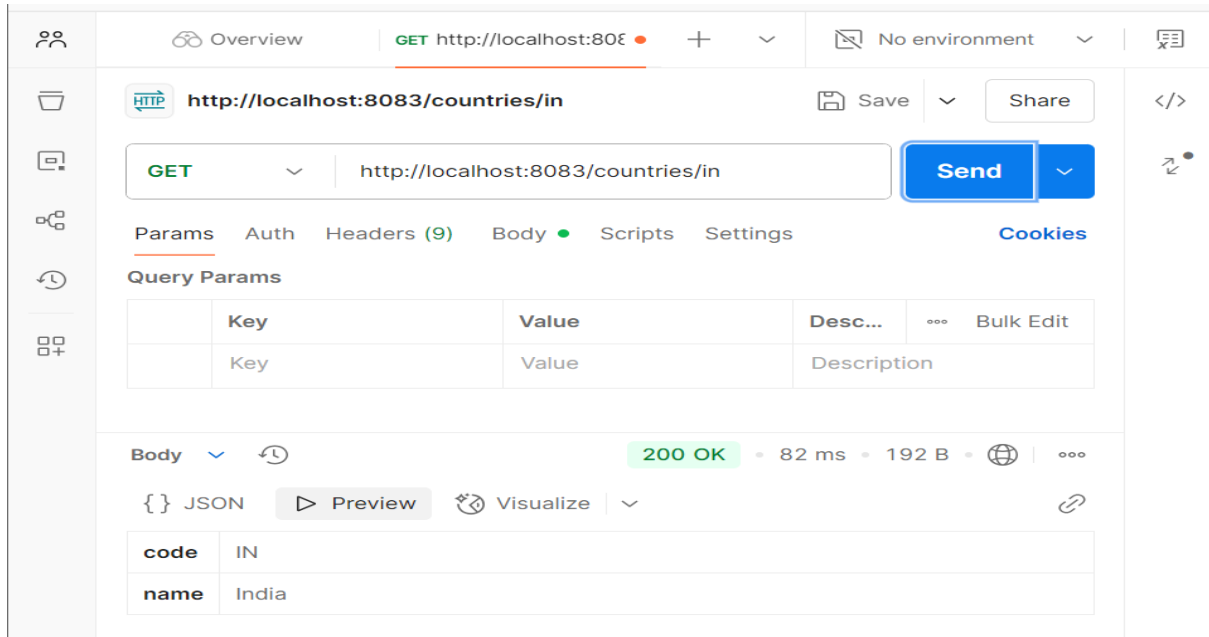
OUTPUT :



The screenshot shows an IDE with the file `SpringLearnApplication.java` open. The code defines a `SpringLearnApplication` class with a `main` method that runs the application and a `displayCountry` method that interacts with the `Country` service. The terminal output shows the following:

```
Description:
Web server failed to start. Port 8083 was already in use.
Action:
Identify and stop the process that's listening on port 8083 or configure this application to listen on another port.

[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 5.063 s
[INFO] Finished at: 2025-07-12T21:40:43+05:30
[INFO] -----
PS C:\Users\VINOTHKUMAR\Downloads\spring-learn (1)\spring-learn>
```



The screenshot shows a web client interface with the following details:

- URL: `http://localhost:8083/countries/in`
- Method: `GET`
- Status: `200 OK`
- Response Time: `82 ms`
- Response Size: `192 B`
- Response Type: `JSON`
- Response Body:

code	value
IN	
name	India

Create authentication service that returns JWT

spring-learn\spring-

learn\src\main\java\com\cognizant\spring_learn\config\SecurityConfig.java

```
package com.cognizant.spring_learn.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.web.SecurityFilterChain;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;

@Configuration
@EnableWebSecurity

public class SecurityConfig {

    @Bean

    public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

        http.csrf(csrf -> csrf.disable())

        .authorizeHttpRequests(auth ->

            auth.requestMatchers("/authenticate").permitAll()

            .anyRequest().authenticated()

        )

        .httpBasic();

        return http.build();

    }

}
```

spring-learn\spring-

learn\src\main\java\com\cognizant\spring_learn\controller\AuthenticationController.java

```
package com.cognizant.spring_learn.controller;

import com.cognizant.spring_learn.util.JwtUtil;
```

```

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*
import java.util.Base64;

@RestController

public class AuthenticationController {

    @Autowired

    private JwtUtil jwtUtil;

    @GetMapping("/authenticate")

    public ResponseEntity<?> authenticate(@RequestHeader("Authorization") String
authHeader) {

        // Decode Basic Auth

        String[] parts = authHeader.split(" ");

        if (parts.length != 2 || !parts[0].equals("Basic")) {

            return ResponseEntity.badRequest().body("Invalid Authorization Header");

        }

        String decoded = new String(Base64.getDecoder().decode(parts[1]));

        String[] credentials = decoded.split(":", 2);

        if (credentials.length != 2 || !credentials[0].equals("user") ||
!credentials[1].equals("pwd")) {

            return ResponseEntity.status(401).body("Invalid Credentials");

        }

        String token = jwtUtil.generateToken(credentials[0]);

        return ResponseEntity.ok().body("{\"token\":\"" + token + "\"}");

    }

}

```

spring-learn\spring-learn\src\main\java\com\cognizant\spring_learn\util\JwtUtil.java

```

package com.cognizant.spring_learn.util;

import io.jsonwebtoken.Jwts;

```

```

import io.jsonwebtoken.SignatureAlgorithm;

import org.springframework.stereotype.Component;

import java.util.Date;

@Component

public class JwtUtil {

    private String secretKey = "mySecretKey";

    public String generateToken(String username) {

        return Jwts.builder()

            .setSubject(username)

            .setIssuedAt(new Date(System.currentTimeMillis()))

            .setExpiration(new Date(System.currentTimeMillis() + 1000 * 60 * 60))

            .signWith(SignatureAlgorithm.HS256, secretKey)

            .compact();

    }

}

```

OUTPUT :

The screenshot shows an IDE with the following components:

- EXPLORER:** Displays the project structure for 'SPRING-LEARN (1)'. The 'controller' folder is expanded, showing 'AuthenticationController.java' selected.
- Editor:** Shows the code for 'AuthenticationController.java'. The 'authenticate' method is visible, which returns a 'ResponseEntity' containing a token.
- TERMINAL:** Shows the output of a Maven build command. The output indicates that the project was installed successfully.

```

[INFO] --- install:3.1.4:install (default-install) @ spring-learn ---
[INFO] Installing C:\Users\VINOTHKUMAR\Downloads\spring-learn (1)\spring-learn\pom.xml to C:\Users\VINOTHKUMAR\.m2\repository\com\cognizant\spring-learn\0.0.1-SNAPSHOT\spring-learn-0.0.1-SNAPSHOT.jar
[INFO] BUILD SUCCESS
[INFO] Total time: 20.767 s
[INFO] Finished at: 2025-07-12T22:10:40+05:30
[INFO]
VINOTHKUMAR@DESKTOP-F3H8064 MINGW64 ~/Downloads/spring-learn (1)/spring-learn (master)
$

```