Spring REST using Spring Boot 3

1. spring-rest-handson

Create a Spring Web Project using Maven

CODE:

```
package com.cognizant.spring_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringLearnApplication {

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

   public static void main(String[] args) {

      LOGGER.info("STARTING SPRING APPLICATION");

      SpringApplication.run(SpringLearnApplication.class, args);

      LOGGER.info("SPRING APPLICATION STARTED");

   }

}
```

OUTPUT:

```
Main] c.c.spring_learn.SpringLearnApplication  : Starting SpringLearnApplication using Java 22.0.1 with PID 21476 (C:\Users\Asus\Desktop\spring-learn\targ
Main] c.c.spring_learn.SpringLearnApplication  : No active profile set, falling back to 1 default profile: "default"
Main] .e.DevToolsPropertyDefaultsPostProcessor : Devtools property defaults active! Set 'spring.devtools.add-properties' to 'false' to disable
Main] .e.DevToolsPropertyDefaultsPostProcessor : For additional web related logging consider setting the 'logging.level.web' property to 'DEBUG'
Main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port 8080 (http)
Main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
Main] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [Apache Tomcat/10.1.42]
Main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
Main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 968 ms
Main] o.s.b.d.a.OptionalLiveReloadServer       : LiveReload server is running on port 35729
Main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8080 (http) with context path '/'
Main] c.c.spring_learn.SpringLearnApplication  : Started SpringLearnApplication in 2.208 seconds (process running for 2.751)
Main] c.c.spring_learn.SpringLearnApplication  : SPRING APPLICATION STARTED
```

Spring Core – Load Country from Spring Configuration XML

CODE:

```
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 + "]";
   }
}
```

```
<?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
                http://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>
```

```
package com.cognizant.spring_learn;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class SpringLearnApplication {
  private static final Logger LOGGER = LoggerFactory.getLogger(SpringLearnApplication.class);
  public static void main(String[] args) {
    LOGGER.info("START");
    displayCountry(); // Make sure this line is present!
    LOGGER.info("END");
  }
  public static void displayCountry() {
    try {
      System.out.println(">>> Loading Spring context...");
      ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");
      System.out.println(">>> Context loaded.");
```

```
        Country country = context.getBean("country", Country.class);
        System.out.println(">>> Bean fetched.");
        LOGGER.debug("Country : {}", country.toString());
        System.out.println(">>> Country = " + country);
      } catch (Exception e) {
        e.printStackTrace(); // log full error to console
        LOGGER.error("Exception occurred while loading country bean", e);
      }
   }
}
```

OUTPUT:

```
<terminated> SpringLearnApplication [Java Application] C:\Users\Asus\.p2\pool\plugins\org.eclipse.justj.openjdk.hots
23:50:40.723 [main] INFO com.cognizant.spring_learn.SpringLearnApplication -- START
>>> Loading Spring context...
>>> Context loaded.
>>> Bean fetched.
>>> Country = Country [code=IN, name=India]
23:50:41.001 [main] INFO com.cognizant.spring_learn.SpringLearnApplication -- END
```

2. spring-rest-handson

Hello World RESTful Web Service

CODE:

spring.application.name=spring-learn

server.port=8083

logging.level.com.cognizant.spring_learn=DEBUG

---

package com.cognizant.spring_learn.controller;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

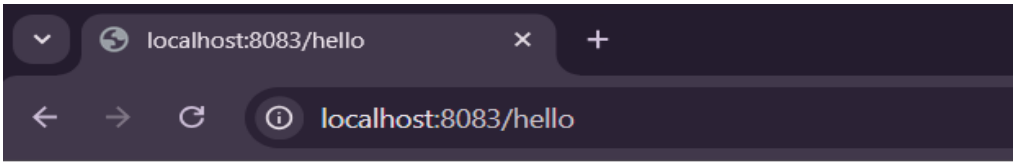
```
import org.springframework.web.bind.annotation.RestController;
@RestController
public class HelloController {
   private static final Logger LOGGER = LoggerFactory.getLogger(HelloController.class);
   @GetMapping("/hello")
   public String sayHello() {
      LOGGER.info("START sayHello()");
      String message = "Hello World!!";
      LOGGER.info("END sayHello()");
      return message;
   }
}
```

```
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) {
      LOGGER.info("START");
      //displayCountry(); // Make sure this line is present!
      SpringApplication.run(SpringLearnApplication.class, args);
      LOGGER.info("END");
   }
```

```
    public static void displayCountry() {
        try {
            System.out.println(">>> Loading Spring context...");
            ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");
            System.out.println(">>> Context loaded.");
            Country country = context.getBean("country", Country.class);
            System.out.println(">>> Bean fetched.");
            LOGGER.debug("Country : {}", country.toString());
            System.out.println(">>> Country = " + country);
        } catch (Exception e) {
            e.printStackTrace(); // log full error to console
            LOGGER.error("Exception occurred while loading country bean", e);
        }
    }
}
```

OUTPUT:

```
restartedMain] c.c.spring_learn.SpringLearnApplication  : No active profile set, falling back to 1 default profile: "default"
restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port 8083 (http)
restartedMain] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
restartedMain] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [Apache Tomcat/10.1.42]
restartedMain] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 168 ms
restartedMain] o.s.b.d.a.OptionalLiveReloadServer       : LiveReload server is running on port 35729
restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8083 (http) with context path '/'
restartedMain] c.c.spring_learn.SpringLearnApplication  : Started SpringLearnApplication in 0.277 seconds (process running for 96.32)
restartedMain] .ConditionEvaluationDeltaLoggingListener : Condition evaluation unchanged
restartedMain] c.c.spring_learn.SpringLearnApplication  : END
```



Hello World!!

REST - Country Web Service

CODE:

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

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

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

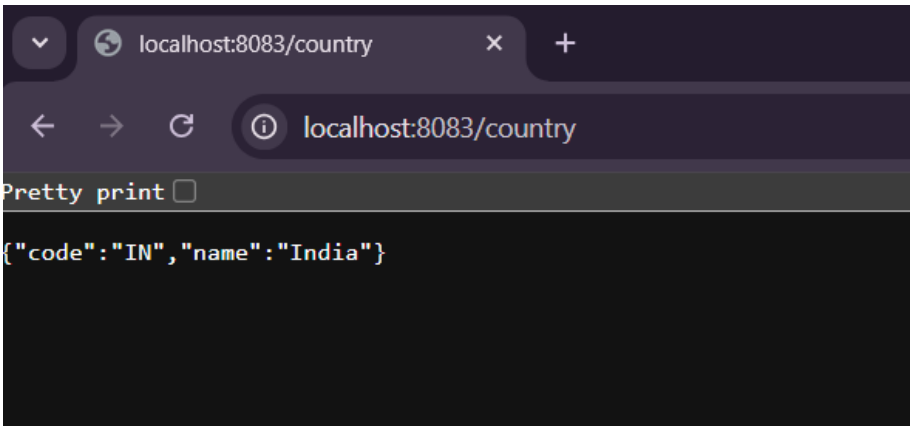
```
<beans xmlns="http://www.springframework.org/schema/beans"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="
        http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans.xsd">
   <bean id="country" class="com.cognizant.spring_learn.model.Country">
      <property name="code" value="IN"/>
      <property name="name" value="India"/>
   </bean>
</beans>
```

```
package com.cognizant.spring_learn.controller;
import com.cognizant.spring_learn.model.Country;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
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()");
      ClassPathXmlApplicationContext context = new ClassPathXmlApplicationContext("country.xml");
      Country country = context.getBean("country", Country.class);
      LOGGER.info("END getCountryIndia()");
      context.close();
      return country;
   }
```

}

OUTPUT:

```
arn] [  restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port 8083 (http)
arn] [  restartedMain] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
arn] [  restartedMain] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [Apache Tomcat/10.1.42]
arn] [  restartedMain] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
arn] [  restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 247 ms
arn] [  restartedMain] o.s.b.d.a.OptionalLiveReloadServer       : LiveReload server is running on port 35729
arn] [  restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8083 (http) with context path '/'
arn] [  restartedMain] c.c.spring_learn.SpringLearnApplication  : Started SpringLearnApplication in 0.368 seconds (process running for 678.394)
arn] [  restartedMain] .ConditionEvaluationDeltaLoggingListener : Condition evaluation unchanged
arn] [  restartedMain] c.c.spring_learn.SpringLearnApplication  : END
arn] [nio-8083-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring DispatcherServlet 'dispatcherServlet'
arn] [nio-8083-exec-1] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
arn] [nio-8083-exec-1] o.s.web.servlet.DispatcherServlet        : Completed initialization in 1 ms
arn] [nio-8083-exec-1] c.c.s.controller.CountryController       : START getCountryIndia()
arn] [nio-8083-exec-1] c.c.s.controller.CountryController       : END getCountryIndia()
```



REST - Get country based on country code

CODE:

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

public class Country {

   private String code;

   private String name;

   public Country() {}

   public Country(String code, String name) {

      this.code = code;

      this.name = name;

   }

   public String getCode() {

      return code;
```

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

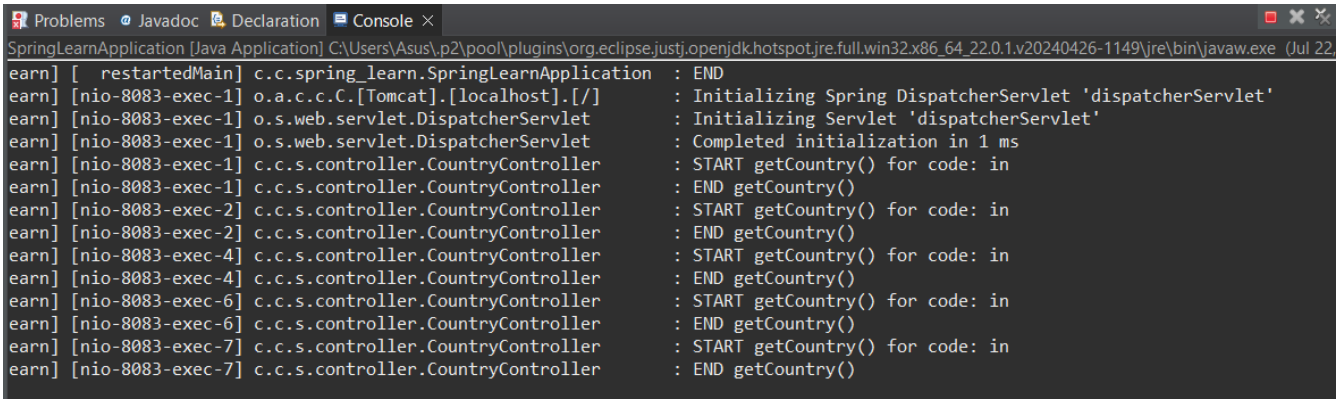
```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="
      http://www.springframework.org/schema/beans
      http://www.springframework.org/schema/beans/spring-beans.xsd">
    <bean id="countryList" class="java.util.ArrayList">
    <constructor-arg>
      <list>
        <bean class="com.cognizant.spring_learn.model.Country">
          <property name="code" value="IN"/>
          <property name="name" value="India"/>
        </bean>
```

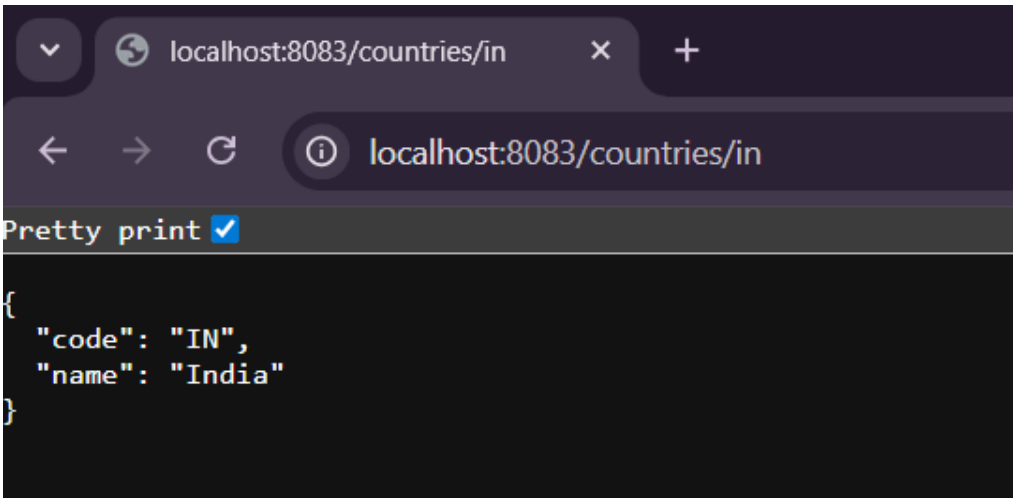
```
        <bean class="com.cognizant.spring_learn.model.Country">
            <property name="code" value="US"/>
            <property name="name" value="United States"/>
        </bean>
        <bean class="com.cognizant.spring_learn.model.Country">
            <property name="code" value="JP"/>
            <property name="name" value="Japan"/>
        </bean>
      </list>
   </constructor-arg>
</bean>
</beans>
```

```
package com.cognizant.spring_learn.controller;
import com.cognizant.spring_learn.model.Country;
import com.cognizant.spring_learn.service.CountryService;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
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);
   @Autowired
   private CountryService countryService;
```

```
    @GetMapping("/countries/{code}")
    public Country getCountry(@PathVariable String code) {
        LOGGER.info("START getCountry() for code: {}", code);
        Country country = countryService.getCountry(code);
        LOGGER.info("END getCountry()");
        return country;
    }
}
```

OUTPUT:





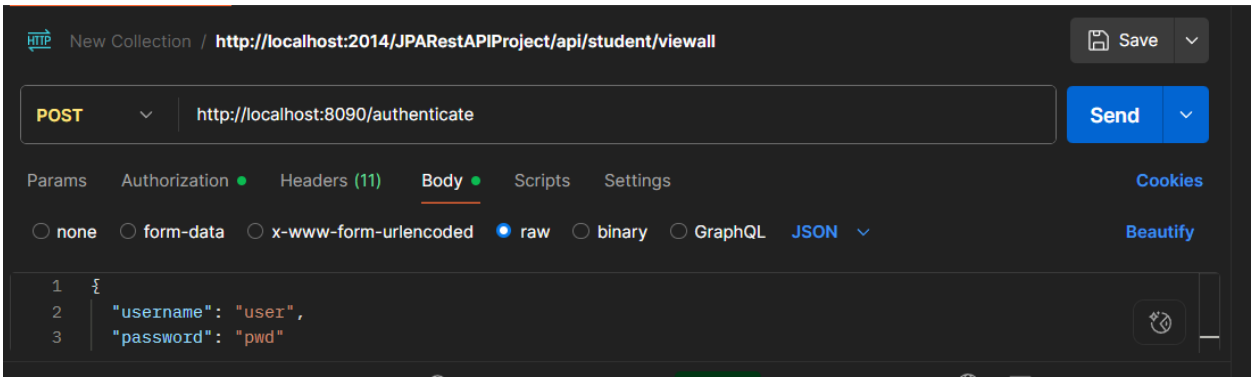
5. JWT-handson

Create authentication service that returns JWT

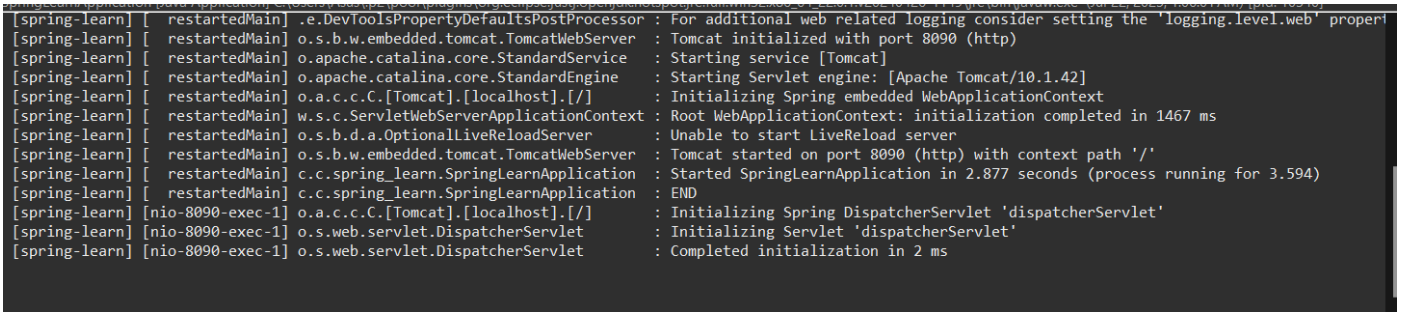
CODE:

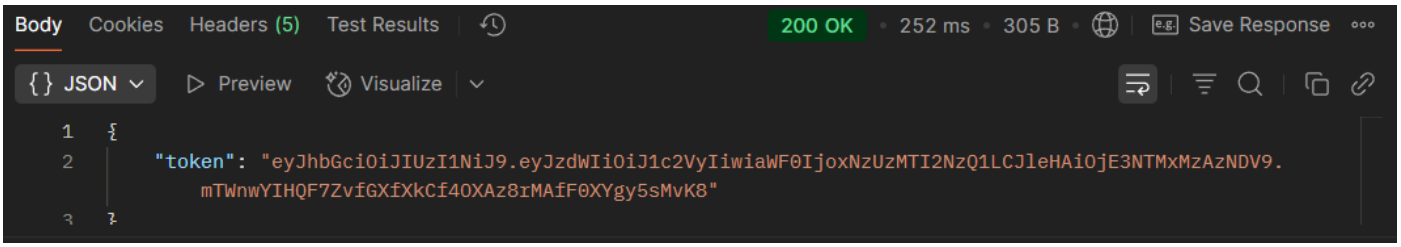
```
package com.cognizant.spring_learn.controller;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

import java.util.Base64;

import java.util.Date;

import java.util.HashMap;

import java.util.Map;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import io.jsonwebtoken.security.Keys;

import java.security.Key;

@RestController

public class AuthenticationController {

   private static final Key key = Keys.secretKeyFor(SignatureAlgorithm.HS256);

   @PostMapping("/authenticate")

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

      // Log start

      System.out.println("AuthenticationController -- START");

      if (authHeader != null && authHeader.startsWith("Basic ")) {

         // Decode the Base64 encoded credentials

         String base64Credentials = authHeader.substring("Basic ".length());

         String credentials = new String(Base64.getDecoder().decode(base64Credentials));

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

         String username = values[0];

         String password = values[1];
```

```
        // Hardcoded check (you can replace this with DB or service check)
        if (username.equals("user") && password.equals("pwd")) {
            String jwt = Jwts.builder()
                    .setSubject(username)
                    .setIssuedAt(new Date())
                    .setExpiration(new Date(System.currentTimeMillis() + 60 * 60 * 1000)) // 1 hour
                    .signWith(SignatureAlgorithm.HS256, key) // ✅ Correct method overload
                    .compact();
            Map<String, String> response = new HashMap<>();
            response.put("token", jwt);
            // Log end
            System.out.println("AuthenticationController -- END");
            return ResponseEntity.ok(response);
        }
    }
    // Log end
    System.out.println("AuthenticationController -- END");
    return ResponseEntity.status(HttpStatus.UNAUTHORIZED).body("Invalid credentials");
  }
}
```



OUTPUT:


[spring-learn] [restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : For additional web related logging consider setting the 'logging.level.web' properl
[spring-learn] [restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port 8090 (http)
[spring-learn] [restartedMain] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
[spring-learn] [restartedMain] o.apache.catalina.core.StandardEngine   : Starting Servlet engine: [Apache Tomcat/10.1.42]
[spring-learn] [restartedMain] o.a.c.c.C.[Tomcat].[localhost].[/]   : Initializing Spring embedded WebApplicationContext
[spring-learn] [restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1467 ms
[spring-learn] [restartedMain] o.s.b.d.a.OptionalLiveReloadServer   : Unable to start LiveReload server
[spring-learn] [restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8090 (http) with context path '/'
[spring-learn] [restartedMain] c.c.spring_learn.SpringLearnApplication : Started SpringLearnApplication in 2.877 seconds (process running for 3.594)
[spring-learn] [restartedMain] c.c.spring_learn.SpringLearnApplication : END
[spring-learn] [nio-8090-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]   : Initializing Spring DispatcherServlet 'dispatcherServlet'
[spring-learn] [nio-8090-exec-1] o.s.web.servlet.DispatcherServlet   : Initializing Servlet 'dispatcherServlet'
[spring-learn] [nio-8090-exec-1] o.s.web.servlet.DispatcherServlet   : Completed initialization in 2 ms


Body
Cookies
Headers (5)
Test Results
200 OK
252 ms
305 B
Save Response
{} JSON
Preview
Visualize
1 {
2 "token": "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ1c2VyIiwiaWF0IjoxNzUzMTI2NzQ1LCJleHAiOjE3NTMxMzAzNDV9.
mTWnwYIHQF7ZvfGXfXkCf4OXAz8rMAfF0XYgy5sMvK8"
3 }