**WEEK 4 – ASSIGNMENT**

**EXERCISE 1: Create a Spring Web Project using Maven   
Xml**

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

**Java**

Package com.cognizant.springlearn;

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

Public class SpringLearnApplication {

Public static void main(String[] args) {

System.out.println(“SpringLearnApplication started...”);

SpringApplication.run(SpringLearnApplication.class, args);

}

@RestController

Class HelloController {

@GetMapping(“/”)

Public String welcomeMessage() {

Return “Welcome to Spring Learn!”;

}

}

}

**Output:**

Welcome to Spring Learn!

Spring rest (All hands on):

**(1) Display Employee List and Edit Employee form using RESTful Web Service**

**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="employeeList" class="java.util.ArrayList">

<constructor-arg>

<list>

<bean class="com.cognizant.employee.Employee">

<property name="id" value="1" />

<property name="name" value="John" />

<property name="salary" value="50000" />

<property name="permanent" value="true" />

</bean>

<bean class="com.cognizant.employee.Employee">

<property name="id" value="2" />

<property name="name" value="Alice" />

<property name="salary" value="60000" />

<property name="permanent" value="false" />

</bean>

<bean class="com.cognizant.employee.Employee">

<property name="id" value="3" />

<property name="name" value="David" />

<property name="salary" value="55000" />

<property name="permanent" value="true" />

</bean>

</list>

</constructor-arg>

</bean>

</beans>

**Java**

package com.cognizant.employee;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.context.ApplicationContext;

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

import java.util.\*;

@SpringBootApplication

public class SpringEmployeeApplication {

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

}

public static class Employee {

private int id;

private String name;

private double salary;

private boolean permanent;

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 double getSalary() { return salary; }

public void setSalary(double salary) { this.salary = salary; }

public boolean isPermanent() { return permanent; }

public void setPermanent(boolean permanent) { this.permanent = permanent; }

}

@RestController

@RequestMapping("/employees")

public class EmployeeController {

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

@SuppressWarnings("unchecked")

List<Employee> employees = (List<Employee>) context.getBean("employeeList");

@GetMapping

public List<Employee> getAllEmployees() {

return employees;

}

@GetMapping("/{id}")

public Employee getEmployeeById(@PathVariable int id) {

return employees.stream()

.filter(e -> e.getId() == id)

.findFirst()

.orElseThrow(() -> new RuntimeException("Employee not found"));

}

}

}

**Output:**

**JSON**

[

{

“id”: 1,

“name”: “John”,

“salary”: 50000.0,

“permanent”: true

},

{

“id”: 2,

“name”: “Alice”,

“salary”: 60000.0,

“permanent”: false

},

{

“id”: 3,

“name”: “David”,

“salary”: 55000.0,

“permanent”: true

}

]

**(2) Create static employee list data using spring xml configuration**

**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">

<!-- Skills -->

<bean id="skill1" class="com.cognizant.model.Skill">

<property name="id" value="1" />

<property name="name" value="Java" />

</bean>

<bean id="skill2" class="com.cognizant.model.Skill">

<property name="id" value="2" />

<property name="name" value="SQL" />

</bean>

<!-- Departments -->

<bean id="dept1" class="com.cognizant.model.Department">

<property name="id" value="1" />

<property name="name" value="IT" />

</bean>

<bean id="dept2" class="com.cognizant.model.Department">

<property name="id" value="2" />

<property name="name" value="HR" />

</bean>

<!-- Employees -->

<bean id="employeeList" class="java.util.ArrayList">

<constructor-arg>

<list>

<bean class="com.cognizant.model.Employee">

<property name="id" value="1" />

<property name="name" value="John" />

<property name="salary" value="50000" />

<property name="permanent" value="true" />

<property name="department" ref="dept1" />

<property name="skills">

<list>

<ref bean="skill1"/>

<ref bean="skill2"/>

</list>

</property>

</bean>

<bean class="com.cognizant.model.Employee">

<property name="id" value="2" />

<property name="name" value="Alice" />

<property name="salary" value="60000" />

<property name="permanent" value="false" />

<property name="department" ref="dept2" />

<property name="skills">

<list>

<ref bean="skill1"/>

</list>

</property>

</bean>

<bean class="com.cognizant.model.Employee">

<property name="id" value="3" />

<property name="name" value="Bob" />

<property name="salary" value="70000" />

<property name="permanent" value="true" />

<property name="department" ref="dept1" />

<property name="skills">

<list>

<ref bean="skill2"/>

</list>

</property>

</bean>

<bean class="com.cognizant.model.Employee">

<property name="id" value="4" />

<property name="name" value="Eve" />

<property name="salary" value="45000" />

<property name="permanent" value="false" />

<property name="department" ref="dept2" />

<property name="skills">

<list>

<ref bean="skill1"/>

<ref bean="skill2"/>

</list>

</property>

</bean>

</list>

</constructor-arg>

</bean>

</beans>

**Java**

package com.cognizant.employee;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.context.ApplicationContext;

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

import java.util.\*;

@SpringBootApplication

public class SpringEmployeeApplication {

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

}

public static class Department {

private int id;

private String name;

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 static class Skill {

private int id;

private String name;

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 static class Employee {

private int id;

private String name;

private double salary;

private boolean permanent;

private Department department;

private List<Skill> skills;

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 double getSalary() { return salary; }

public void setSalary(double salary) { this.salary = salary; }

public boolean isPermanent() { return permanent; }

public void setPermanent(boolean permanent) { this.permanent = permanent; }

public Department getDepartment() { return department; }

public void setDepartment(Department department) { this.department = department; }

public List<Skill> getSkills() { return skills; }

public void setSkills(List<Skill> skills) { this.skills = skills; }

}

public static class EmployeeDao {

private static List<Employee> EMPLOYEE\_LIST;

public EmployeeDao() {

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

EMPLOYEE\_LIST = (List<Employee>) context.getBean("employeeList");

}

public List<Employee> getAllEmployees() {

return EMPLOYEE\_LIST;

}

}

@RestController

@RequestMapping("/employees")

public class EmployeeController {

EmployeeDao employeeDao = new EmployeeDao();

@GetMapping

public List<Employee> getEmployees() {

return employeeDao.getAllEmployees();

}

@GetMapping("/{id}")

public Employee getEmployeeById(@PathVariable int id) {

return employeeDao.getAllEmployees()

.stream()

.filter(e -> e.getId() == id)

.findFirst()

.orElseThrow(() -> new RuntimeException("Employee not found"));

}

}

}

**Output:**

**JSON**

[

{

“id”: 1,

“name”: “John”,

“salary”: 50000.0,

“permanent”: true,

“department”: {

“id”: 1,

“name”: “IT”

},

“skills”: [

{ “id”: 1, “name”: “Java” },

{ “id”: 2, “name”: “SQL” }

]

},

{

“id”: 2,

“name”: “Alice”,

“salary”: 60000.0,

“permanent”: false,

“department”: {

“id”: 2,

“name”: “HR”

},

“skills”: [

{ “id”: 2, “name”: “SQL” }

]

}

]

**(3) Create REST service to gets all employees**

**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="employeeList" class="java.util.ArrayList">

<constructor-arg>

<list>

<bean class="com.cognizant.spring.SpringLearnApplication$Employee">

<property name="id" value="1"/>

<property name="name" value="John"/>

<property name="salary" value="50000"/>

</bean>

<bean class="com.cognizant.spring.SpringLearnApplication$Employee">

<property name="id" value="2"/>

<property name="name" value="Alice"/>

<property name="salary" value="60000"/>

</bean>

</list>

</constructor-arg>

</bean>

</beans>

**Java**

package com.cognizant.spring;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.Bean;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

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

import java.util.\*;

@SpringBootApplication

public class SpringLearnApplication {

public static void main(String[] args) {

SpringApplication.run(SpringLearnApplication.class, args);

}

public static class Employee {

private int id;

private String name;

private double salary;

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 double getSalary() { return salary; }

public void setSalary(double salary) { this.salary = salary; }

}

public static class EmployeeDao {

private static List<Employee> EMPLOYEE\_LIST;

public EmployeeDao() {

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

EMPLOYEE\_LIST = (List<Employee>) context.getBean("employeeList");

}

public List<Employee> getAllEmployees() {

return EMPLOYEE\_LIST;

}

}

@Service

public static class EmployeeService {

private EmployeeDao employeeDao = new EmployeeDao();

@Transactional

public List<Employee> getAllEmployees() {

return employeeDao.getAllEmployees();

}

}

@RestController

public static class EmployeeController {

EmployeeService employeeService = new EmployeeService();

@GetMapping("/employees")

public List<Employee> getAllEmployees() {

return employeeService.getAllEmployees();

}

}

}

**Output:**

**JSON**

[

{

"id": 1,

"name": "John",

"salary": 50000.0

},

{

"id": 2,

"name": "Alice",

"salary": 60000.0

}

]

**(4) Create REST service for department**

**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="departmentList" class="java.util.ArrayList">

<constructor-arg>

<list>

<bean class="com.cognizant.spring.SpringLearnApplication$Department">

<property name="id" value="1"/>

<property name="name" value="IT"/>

</bean>

<bean class="com.cognizant.spring.SpringLearnApplication$Department">

<property name="id" value="2"/>

<property name="name" value="HR"/>

</bean>

<bean class="com.cognizant.spring.SpringLearnApplication$Department">

<property name="id" value="3"/>

<property name="name" value="Finance"/>

</bean>

</list>

</constructor-arg>

</bean>

</beans>

**Java**

Package com.cognizant.spring;

Import org.springframework.boot.SpringApplication;

Import org.springframework.boot.autoconfigure.SpringBootApplication;

Import org.springframework.context.ApplicationContext;

Import org.springframework.context.support.ClassPathXmlApplicationContext;

Import org.springframework.stereotype.Service;

Import org.springframework.transaction.annotation.Transactional;

Import org.springframework.web.bind.annotation.\*;

Import java.util.List;

@SpringBootApplication

Public class SpringLearnApplication {

Public static void main(String[] args) {

SpringApplication.run(SpringLearnApplication.class, args);

System.out.println(“Spring Boot Department Service is running...”);

}

Public static class Department {

Private int id;

Private String name;

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 static class DepartmentDao {

Private static List<Department> DEPARTMENT\_LIST;

Public DepartmentDao() {

ApplicationContext context = new ClassPathXmlApplicationContext(“department.xml”);

DEPARTMENT\_LIST = (List<Department>) context.getBean(“departmentList”);

}

Public List<Department> getAllDepartments() {

Return DEPARTMENT\_LIST;

}

}

@Service

Public static class DepartmentService {

Private DepartmentDao departmentDao = new DepartmentDao();

@Transactional

Public List<Department> getAllDepartments() {

System.out.println(“Fetching all departments from DepartmentService...”);

Return departmentDao.getAllDepartments();

}

}

@RestController

Public static class DepartmentController {

DepartmentService departmentService = new DepartmentService();

@GetMapping(“/departments”)

Public List<Department> getAllDepartments() {

System.out.println(“GET /departments called in DepartmentController”);

Return departmentService.getAllDepartments();

}

}

}

**Output:**

**JSON**

[

{

“id”: 1,

“name”: “IT”

},

{

“id”: 2,

“name”: “HR”

},

{

“id”: 3,

“name”: “Finance”

}

]

**Logs Output:**

Spring Boot Department Service is running...

GET /departments called in DepartmentController

Fetching all departments from DepartmentService...

**EXERCISE 2: Spring Core – Load SimpleDateFormat from Spring Configuration XML**

**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=”dateFormat” class=”java.text.SimpleDateFormat”>

<constructor-arg value=”dd/MM/yyyy” />

</bean>

</beans>

**Java**

Package com.cognizant.springlearn;

Import java.text.SimpleDateFormat;

Import java.util.Date;

Import org.springframework.context.ApplicationContext;

Import org.springframework.context.support.ClassPathXmlApplicationContext;

Import org.springframework.boot.SpringApplication;

Import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

Public class SpringLearnApplication {

Public static void main(String[] args) {

System.out.println(“SpringLearnApplication started...”);

SpringApplication.run(SpringLearnApplication.class, args);

displayDate();

}

Public static void displayDate() {

ApplicationContext context = new ClassPathXmlApplicationContext(“date-format.xml”);

SimpleDateFormat format = context.getBean(“dateFormat”, SimpleDateFormat.class);

Try {

Date date = format.parse(“31/12/2018”);

System.out.println(“Parsed Date: “ + date);

} catch (Exception e) {

System.out.println(“Error parsing date: “ + e.getMessage());

}

}

}

**Output:**

SpringLearnApplication started...

Parsed Date: Mon Dec 31 00:00:00 IST 2018

**EXERCISE 3: Hello World RESTful Web Service**

**Java**

package com.cognizant.springlearn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

public class SpringLearnApplication {

public static void main(String[] args) {

SpringApplication.run(SpringLearnApplication.class, args);

}

@RestController

class HelloController {

private 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:**

Hello World!!

**Console Log Output:**

INFO ... HelloController: START - sayHello()

INFO ... HelloController: END - sayHello()

**EXERCISE 4: REST – Country Web Service**

**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="in" class="com.cognizant.springlearn.model.Country">

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

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

</bean>

</beans>

**Java**

package com.cognizant.springlearn;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

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;

@SpringBootApplication

public class SpringLearnApplication {

public static void main(String[] args) {

SpringApplication.run(SpringLearnApplication.class, args);

}

public static class Country {

private String code;

private String 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;

}

}

@RestController

class CountryController {

@RequestMapping("/country")

public Country getCountryIndia() {

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

return (Country) context.getBean("in");

}

}

}

**Output:**

**JSON**

{

"code": "IN",

"name": "India"

}

**Console Output:**

:: Spring Boot :: (v3.x.x)

SpringLearnApplication started...

Tomcat started on port(s): 8083 (http)

Started SpringLearnApplication in X.XXX seconds

**EXERCISE 5: REST – Get country based on country code**

**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=”countryList” class=”java.util.ArrayList”>

<constructor-arg>

<list>

<bean class=”com.cognizant.springlearn.controller.CountryController$Country”>

<property name=”code” value=”IN”/>

<property name=”name” value=”India”/>

</bean>

<bean class=”com.cognizant.springlearn.controller.CountryController$Country”>

<property name=”code” value=”US”/>

<property name=”name” value=”United States”/>

</bean>

</list>

</constructor-arg>

</bean>

</beans>

**Java**

Package com.cognizant.springlearn.controller;

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

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

Import org.springframework.web.bind.annotation.PathVariable;

Import java.util.List;

Import java.util.ArrayList;

Import org.springframework.context.ApplicationContext;

Import org.springframework.context.support.ClassPathXmlApplicationContext;

@RestController

Public class CountryController {

Public static class Country {

Private String code;

Private String 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;

}

}

@GetMapping(“/country/{code}”)

Public Country getCountry(@PathVariable String code) {

ApplicationContext context = new ClassPathXmlApplicationContext(“country.xml”);

List<Country> countries = (List<Country>) context.getBean(“countryList”);

For (Country country : countries) {

If (country.getCode().equalsIgnoreCase(code)) {

Return country;

}

}

Return null;

}

}

**Output:**

**JSON**

{

“code”: “IN”,

“name”: “India”

}

**Console Output:**

Content-Type: application/json

Transfer-Encoding: chunked

Date: [current timestamp]

**EXERCISE 6: Create authentication service that returns JWT**

**Xml**

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-security</artifactId>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

**Java**

Package com.cognizant.jwt;

Import org.springframework.boot.SpringApplication;

Import org.springframework.boot.autoconfigure.SpringBootApplication;

Import org.springframework.context.annotation.Bean;

Import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

Import org.springframework.web.bind.annotation.\*;

Import org.springframework.http.\*;

Import javax.servlet.http.HttpServletRequest;

Import java.util.\*;

Import java.util.Base64;

Import java.util.Date;

Import io.jsonwebtoken.\*;

@SpringBootApplication

Public class SpringJwtAuthApplication {

Public static void main(String[] args) {

SpringApplication.run(SpringJwtAuthApplication.class, args);

}

}

@RestController

Class AuthController {

Private final String SECRET\_KEY = “secret123”;

@RequestMapping(value = “/authenticate”, method = RequestMethod.GET)

Public ResponseEntity<Map<String, String>> authenticate(HttpServletRequest request) {

String authHeader = request.getHeader(“Authorization”);

If (authHeader == null || !authHeader.startsWith(“Basic “)) {

Return ResponseEntity.status(HttpStatus.UNAUTHORIZED).build();

}

String base64Credentials = authHeader.substring(“Basic “.length()).trim();

Byte[] credDecoded = Base64.getDecoder().decode(base64Credentials);

String credentials = new String(credDecoded);

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

String username = values[0];

String password = values[1];

If (“user”.equals(username) && “pwd”.equals(password)) {

String token = generateToken(username);

Map<String, String> response = new HashMap<>();

Response.put(“token”, token);

Return ResponseEntity.ok(response);

}

Return ResponseEntity.status(HttpStatus.UNAUTHORIZED).build();

}

Private String generateToken(String username) {

Return Jwts.builder()

.setSubject(username)

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

.setExpiration(new Date(System.currentTimeMillis() + 1000 \* 60 \* 10)) // 10 minutes

.signWith(SignatureAlgorithm.HS256, SECRET\_KEY)

.compact();

}

}

**SecurityConfig.java**

Package com.cognizant.jwt.config;

Import org.springframework.context.annotation.Configuration;

Import org.springframework.security.config.annotation.web.builders.HttpSecurity;

Import org.springframework.security.web.SecurityFilterChain;

Import org.springframework.context.annotation.Bean;

@Configuration

Public class SecurityConfig {

@Bean

Public SecurityFilterChain filterChain(HttpSecurity http) throws Exception {

http

.csrf().disable()

.authorizeRequests()

.anyRequest().permitAll(); // We handle auth ourselves

Return http.build();

}

}

**Output:**

**JSON**

{

“token”: “eyJhbGciOiJIUzI1NiJ9...”

}

Content-Type: application/json

Status: 200 OK