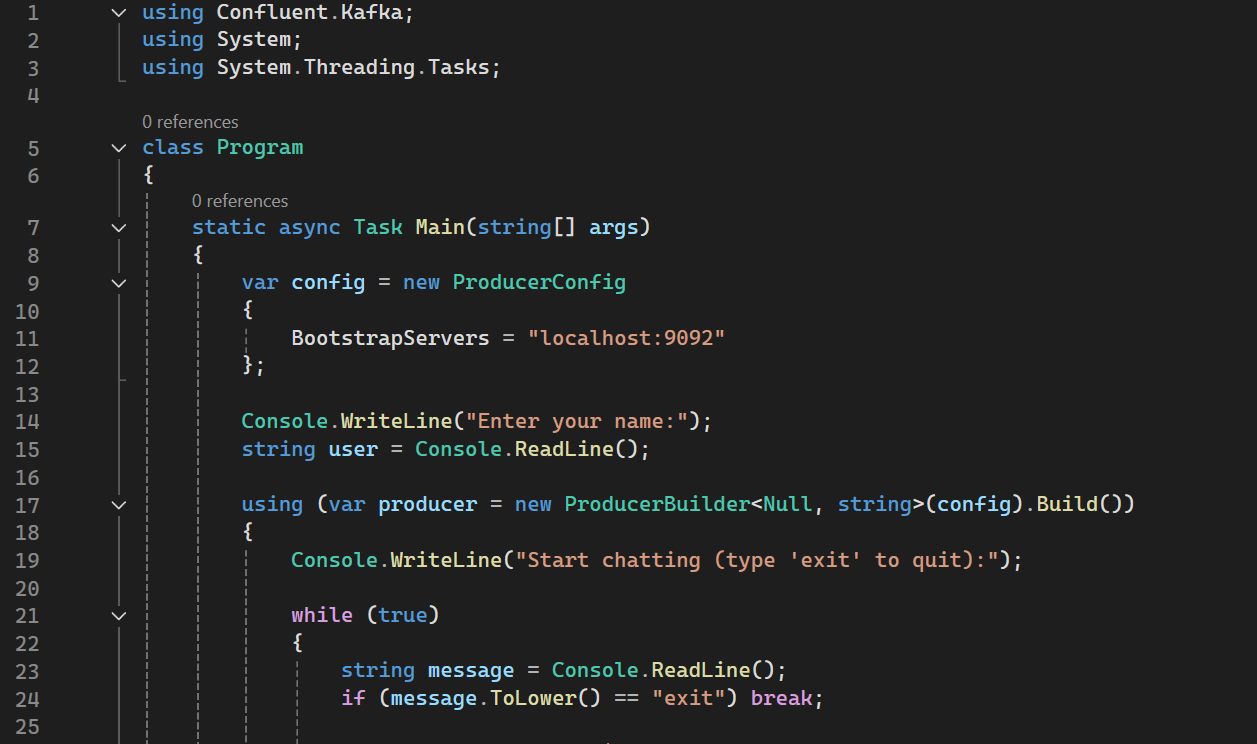
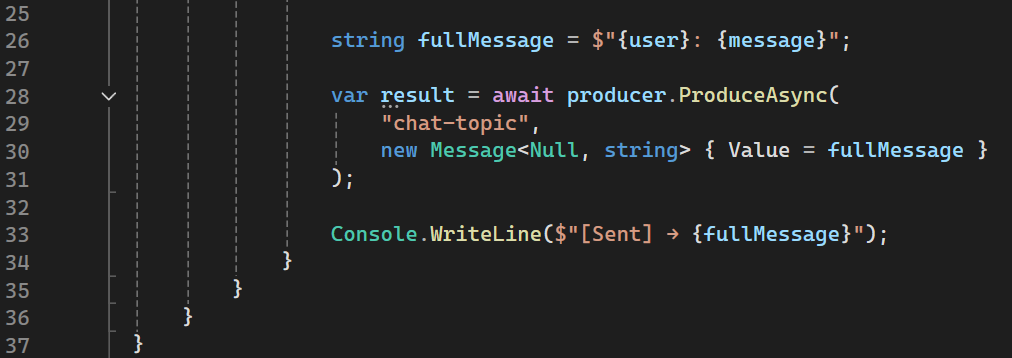
**6.WebApi\_Handson**

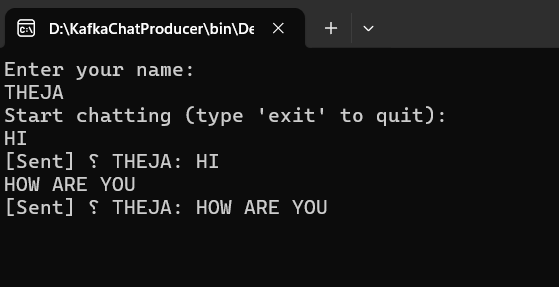
**TASK 1 :** Create a Chat Application which uses Kafka as a streaming platform and consume the chat messages in the command prompt.

**Program.cs**

**OUTPUT:**

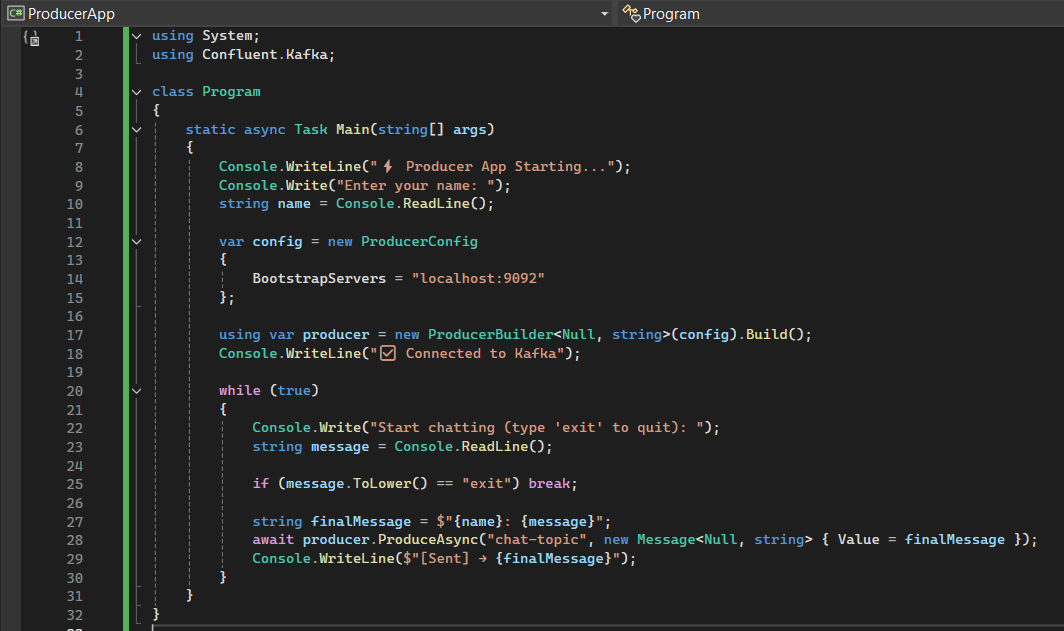
consumes the chat messages in the command prompt.

****

**TASK 2:** Create a Chat Application using C# Windows Application using Kafka and consume the message in different client applications.

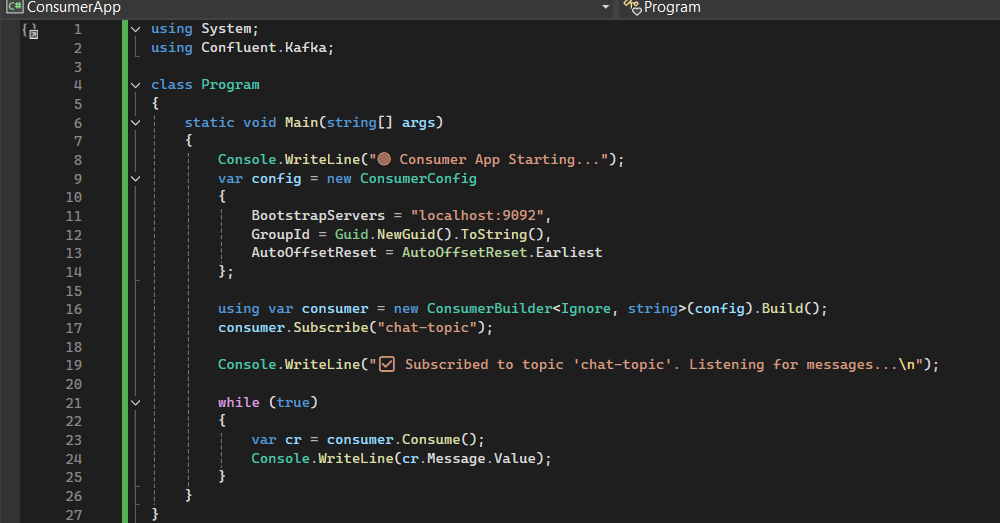
**ProducerApp**

Program.cs

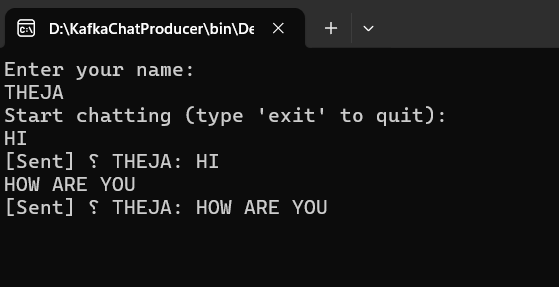


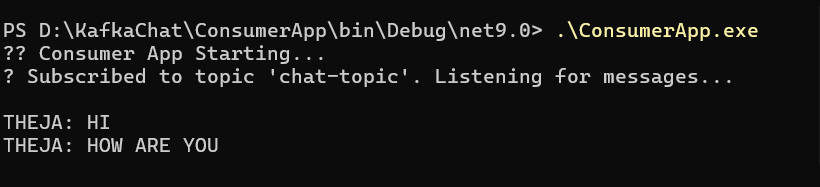
ConsumerApp

Program.cs



**OUTPUT:**

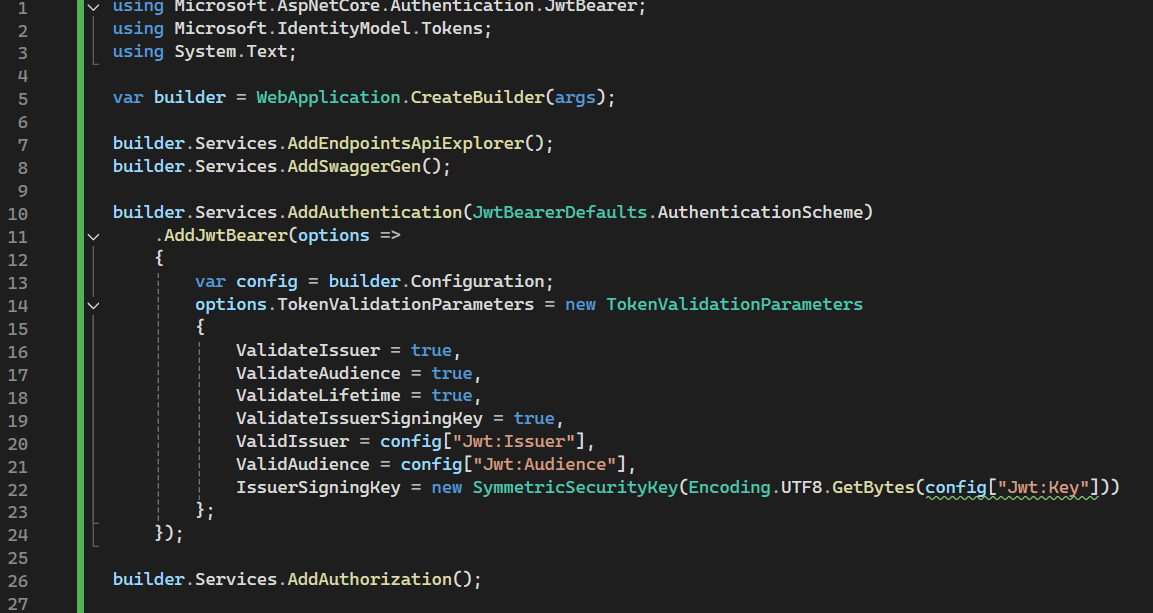
****

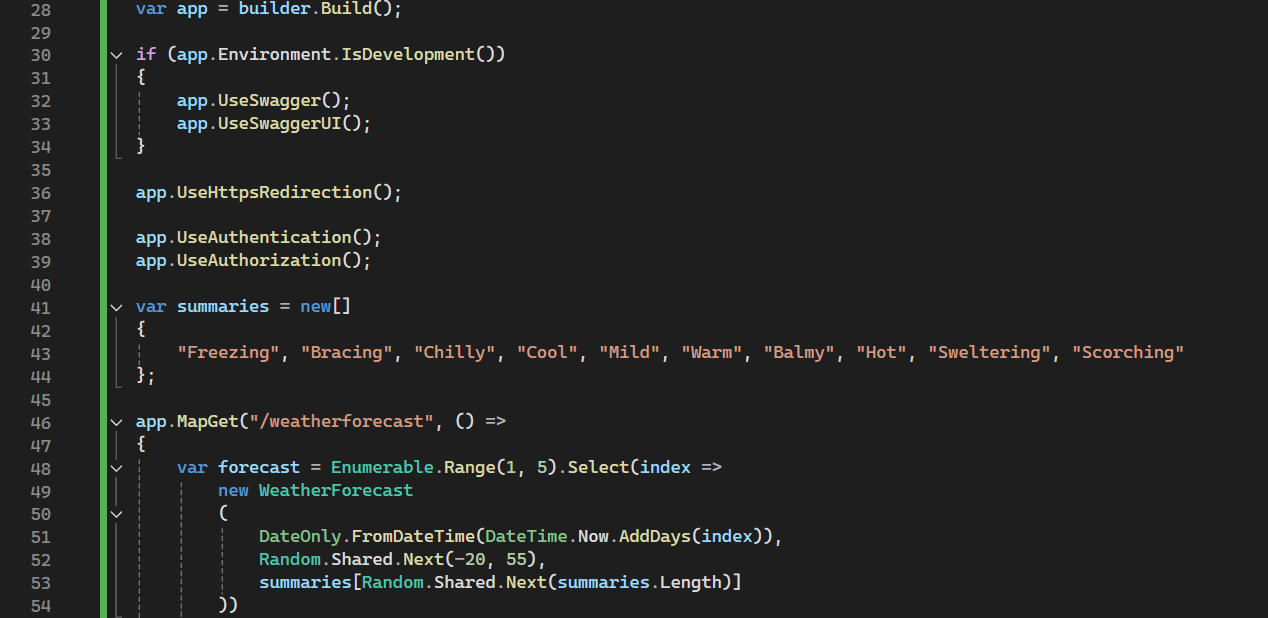
****

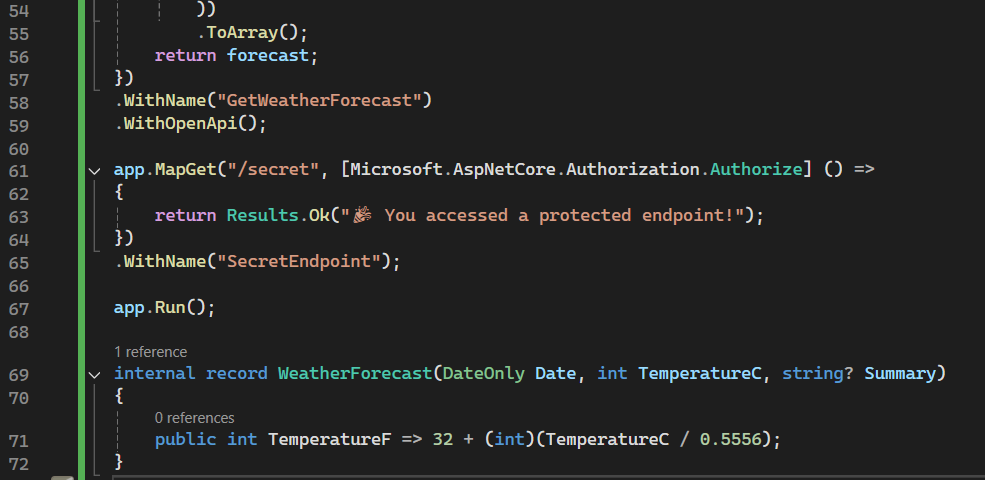
**Authentication and Authorization in ASP.NET Core Web API Microservices Hands-On Exercises**

**TASK 1: Implement JWT Authentication in ASP.NET Core Web API**

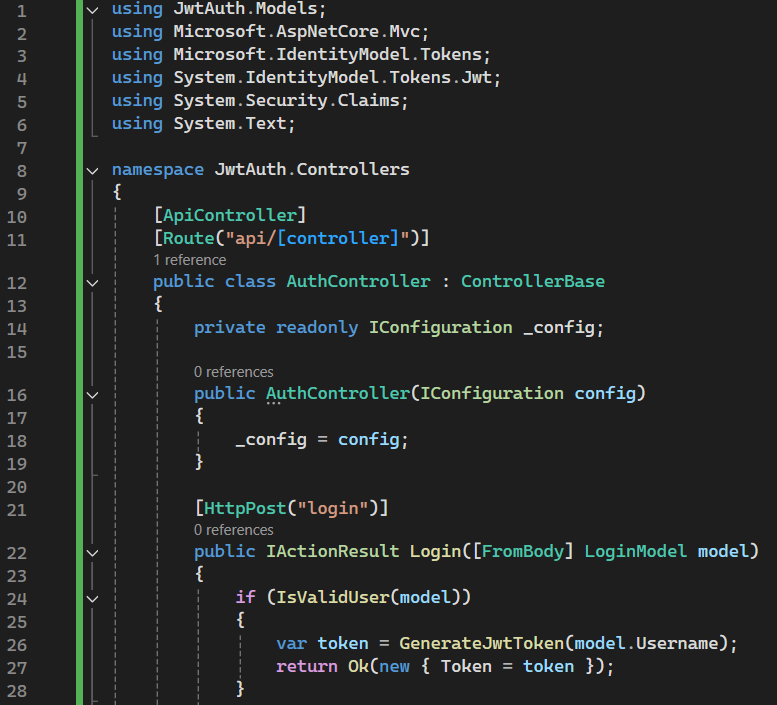
**Program.cs**

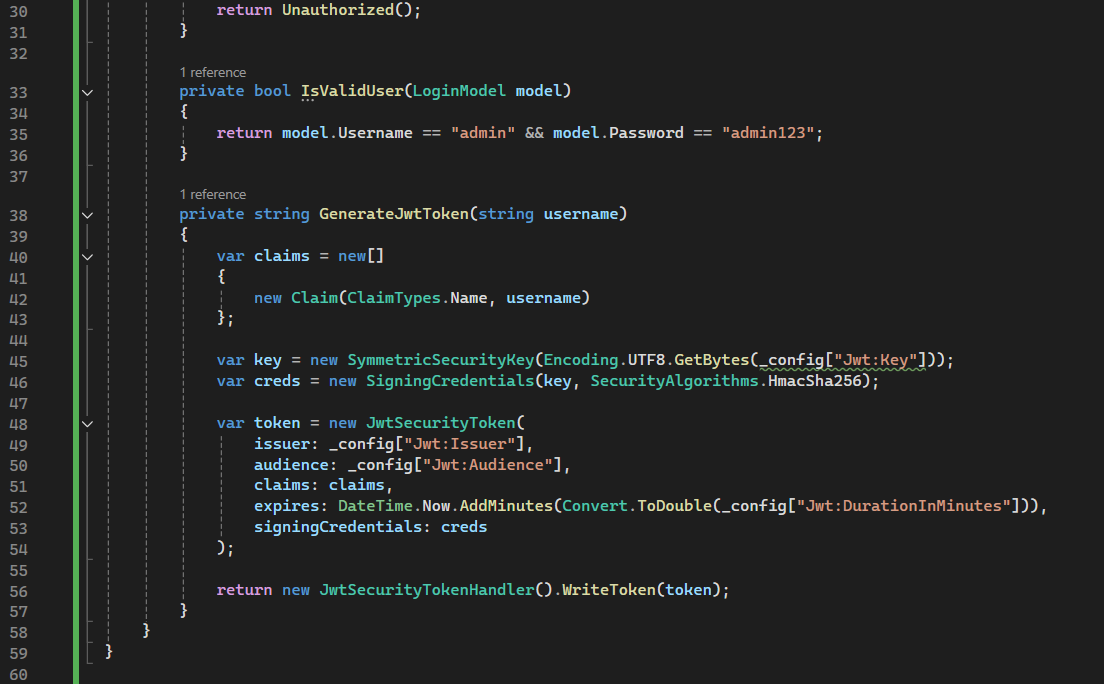
****

****

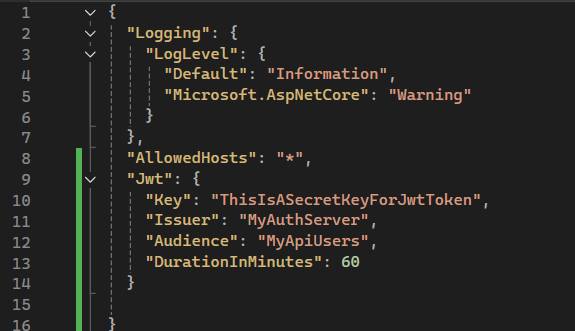
****

**AuthController.cs**

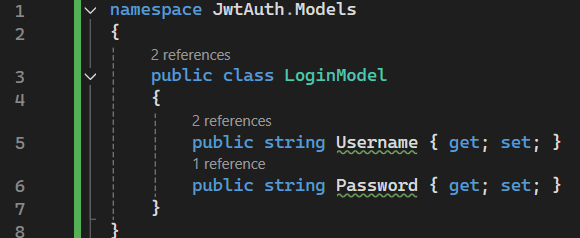
****

****

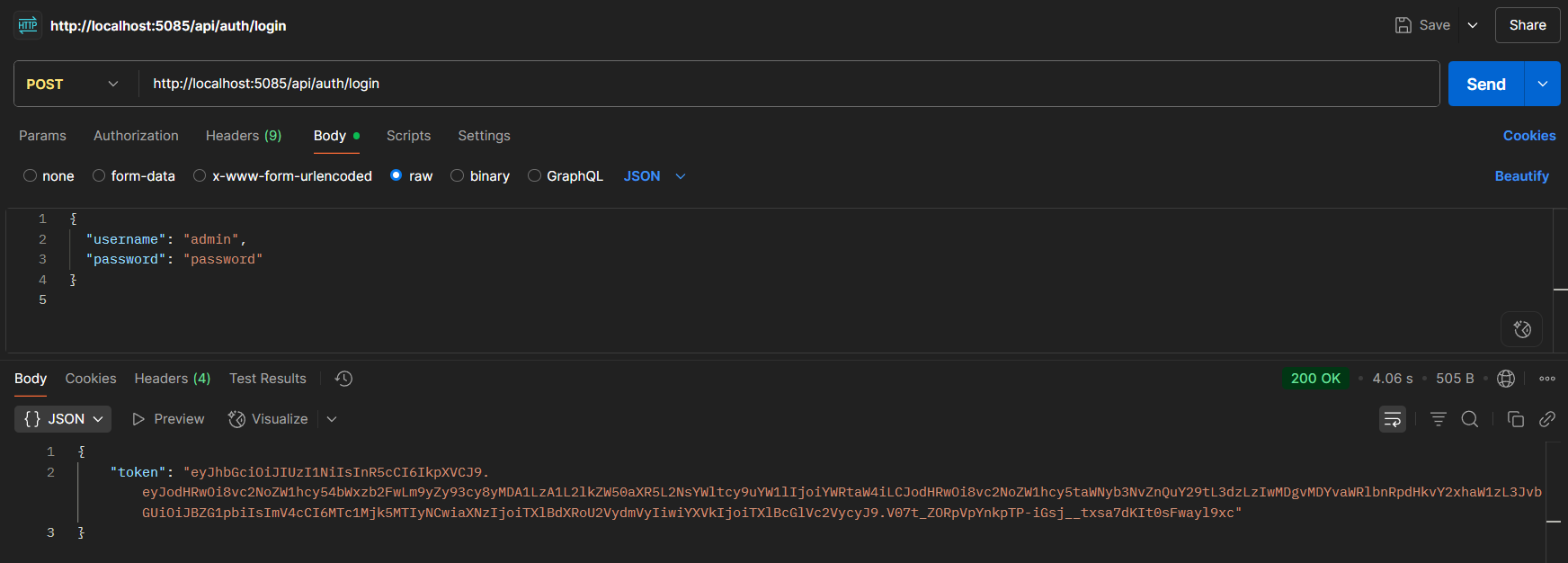
**appsettings.json**

****

**LoginModel.cs**

****

**OUTPUT:**

****

**TASK 2: Secure an API Endpoint Using JWT**

**Scenario:**

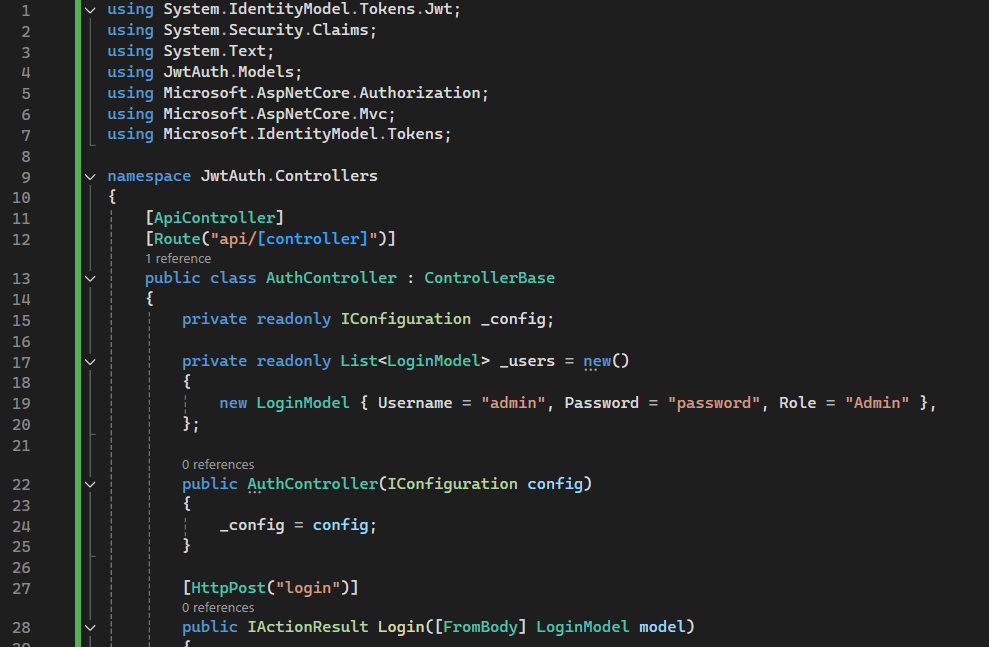
You want to restrict access to a sensitive endpoint using JWT authentication.

**Steps:**

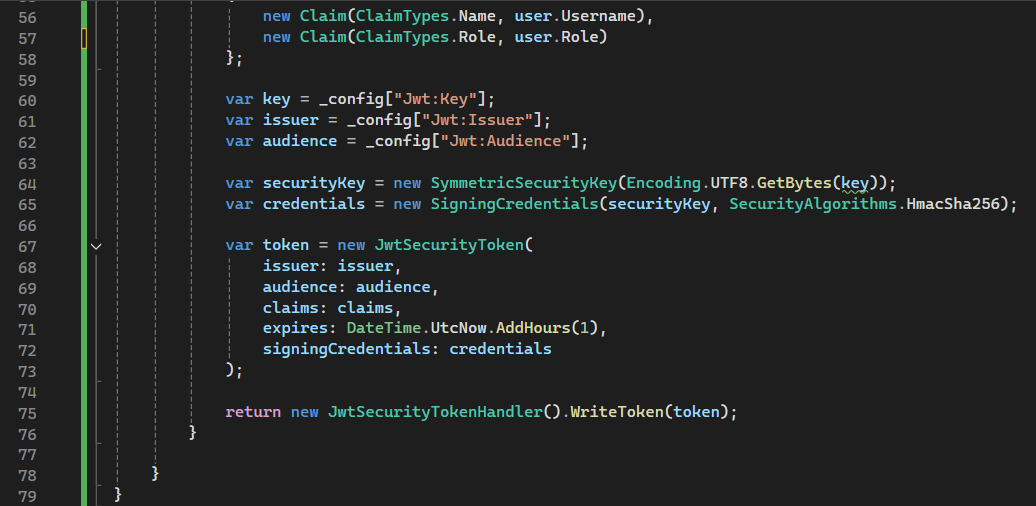
1. Add `[Authorize]` to a controller.

2. Test access with and without a valid token.

Program.cs

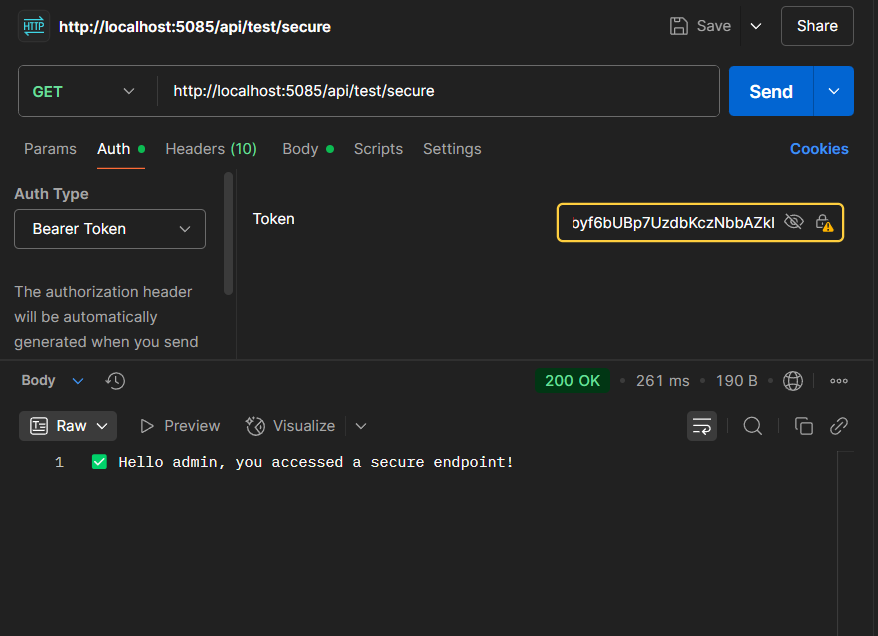
****

****

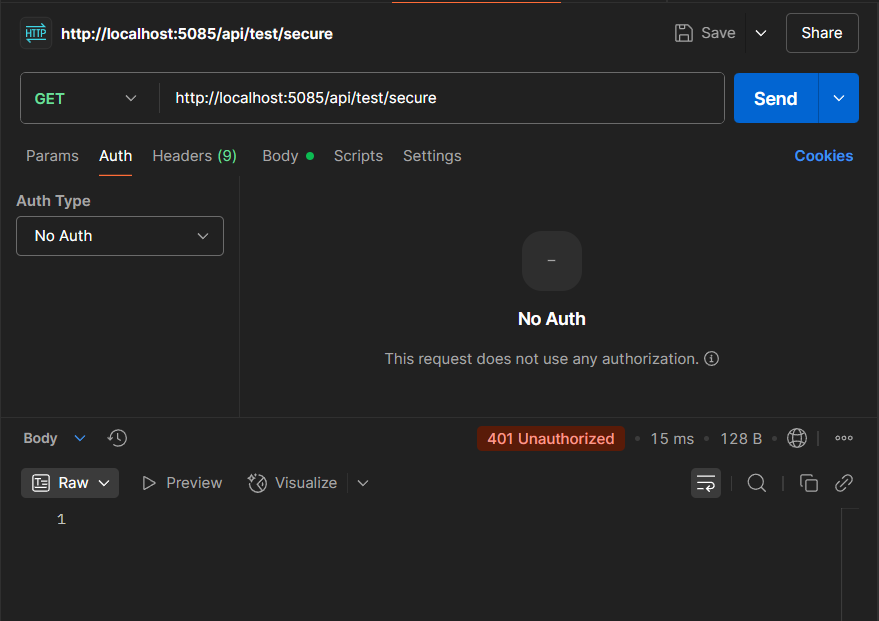
****

**OUTPUT:**

**With Header**

****

**Without header**

****

**TASK 3:** **Add Role-Based Authorization**

**Scenario:**

You want to allow only users with the "Admin" role to access certain endpoints.

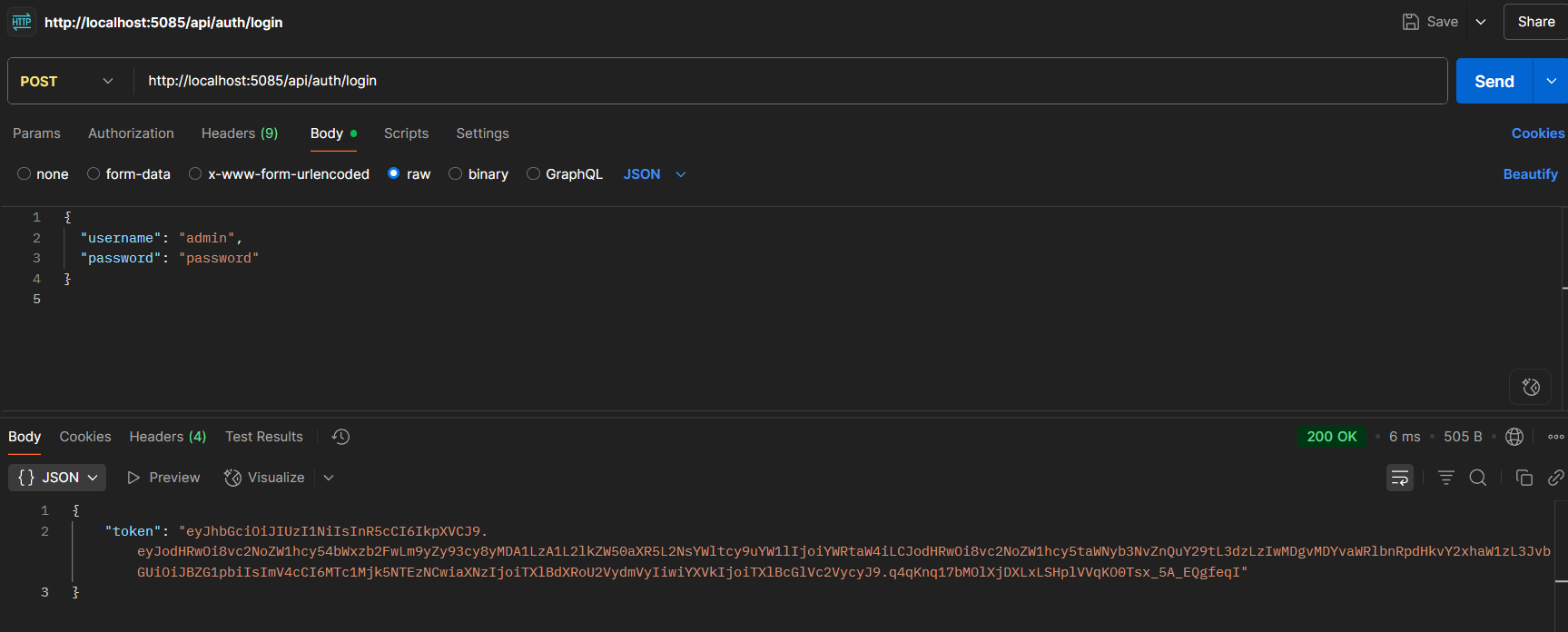
**Steps:**

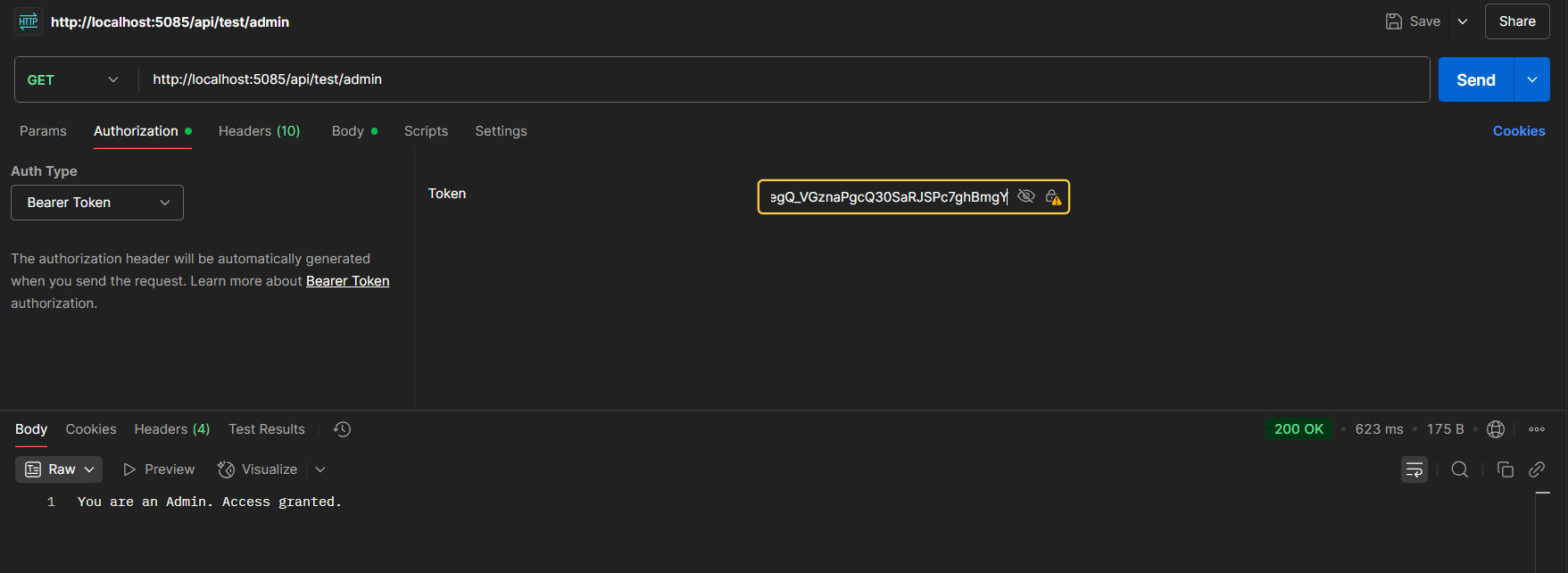
1. Add roles to JWT claims.

2. Use `[Authorize(Roles = "Admin")]`.

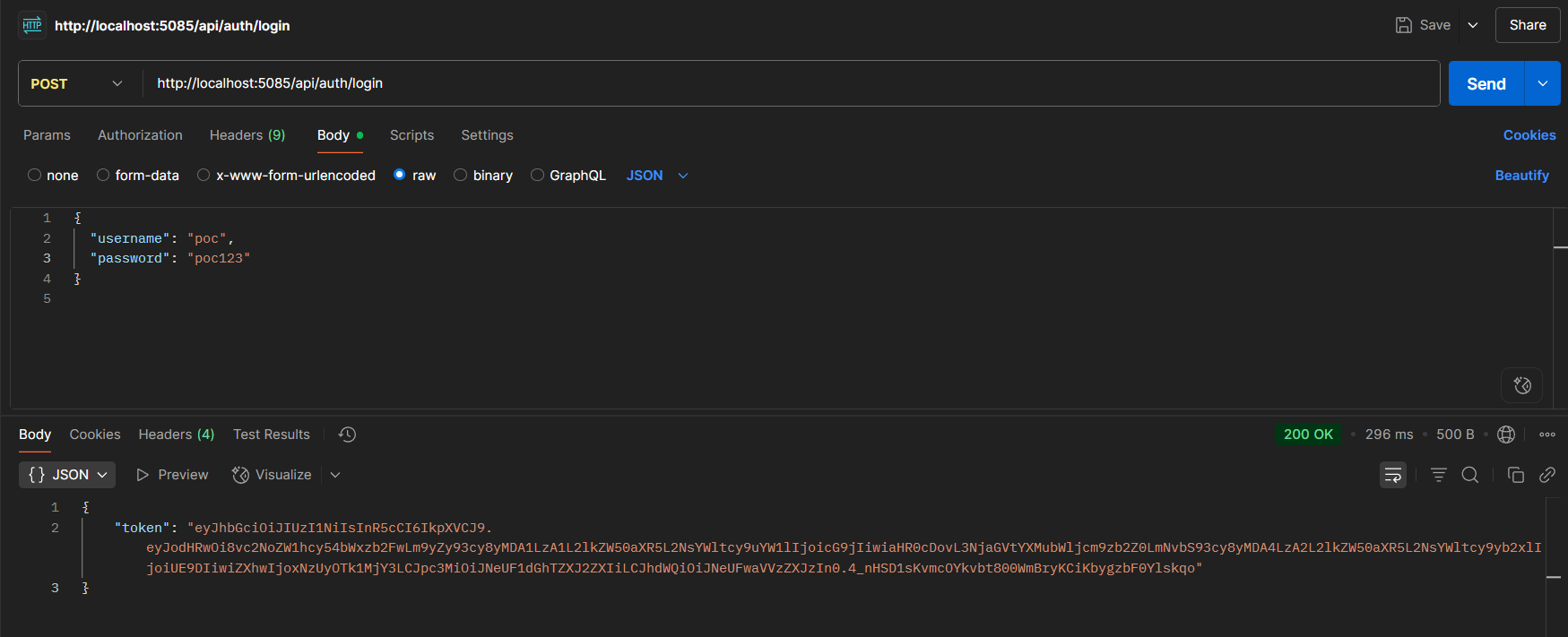
**OUTPUT:**

**i)Admin**

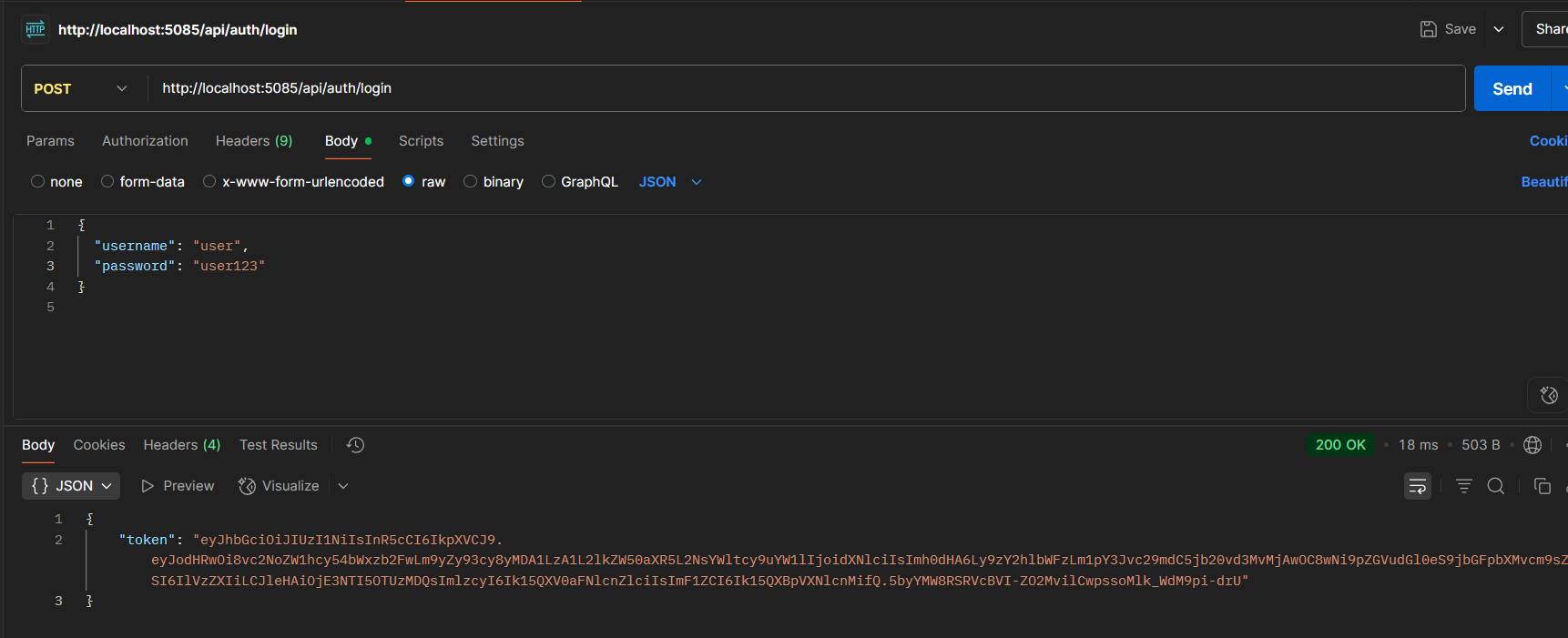
****

****

**ii)poc**

****

**iii)user**

****

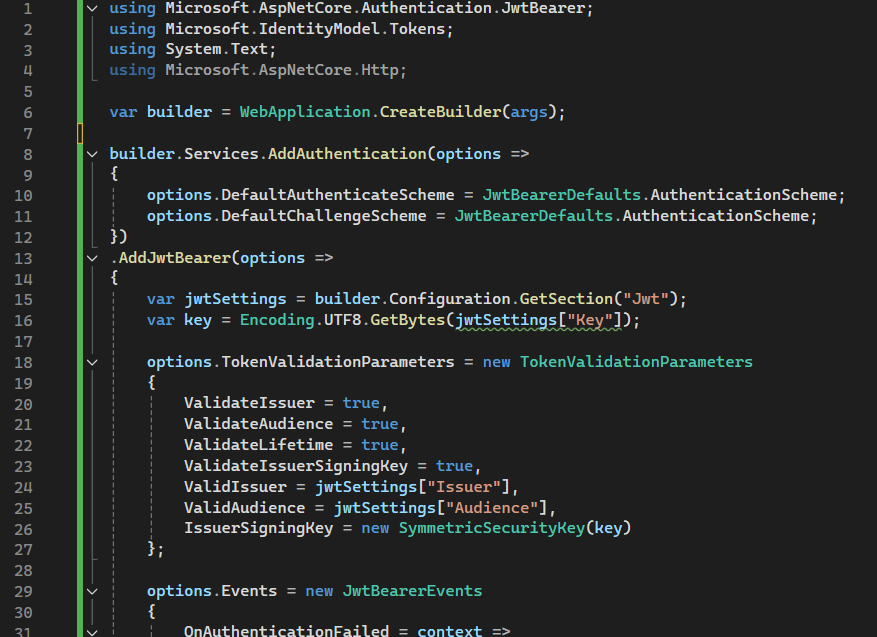
**TASK 4:** Validate JWT Token Expiry and Handle Unauthorized Access Scenario: You want to handle expired or invalid tokens gracefully.

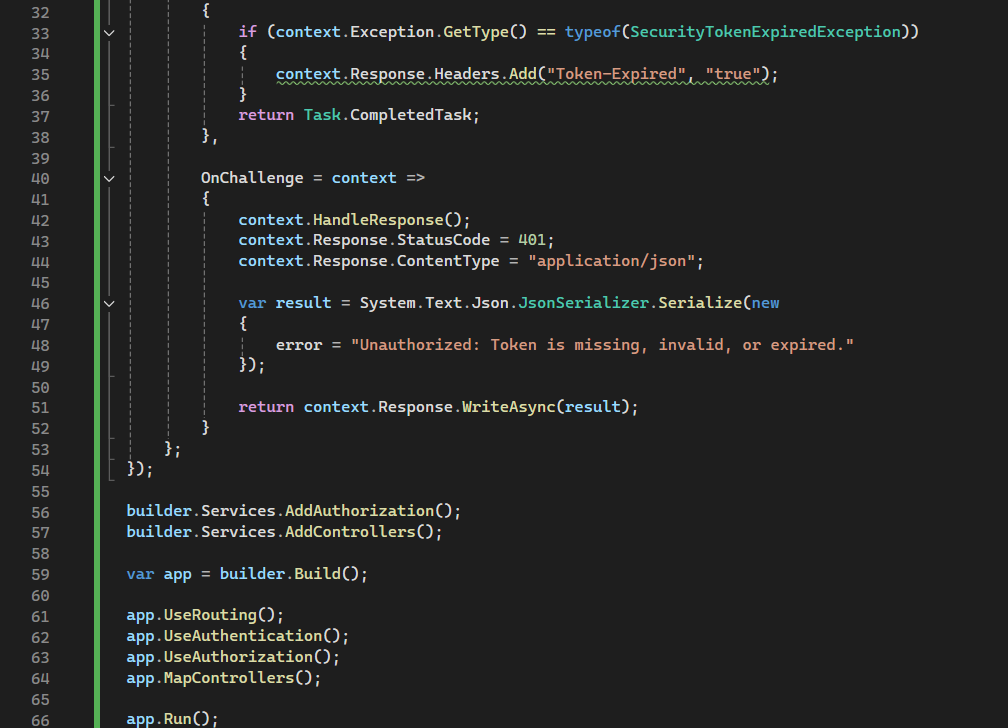
Steps:

1. Configure JWT bearer events.

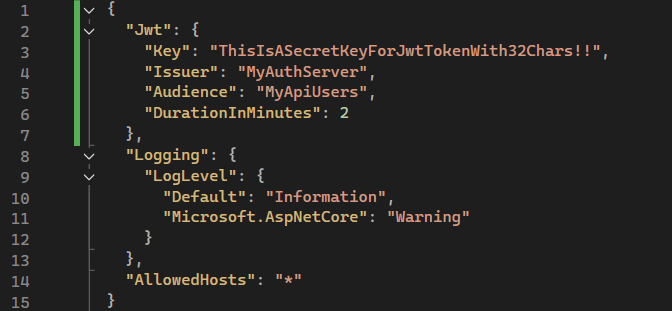
2. Return custom messages for unauthorized access.

**Program.cs**

****

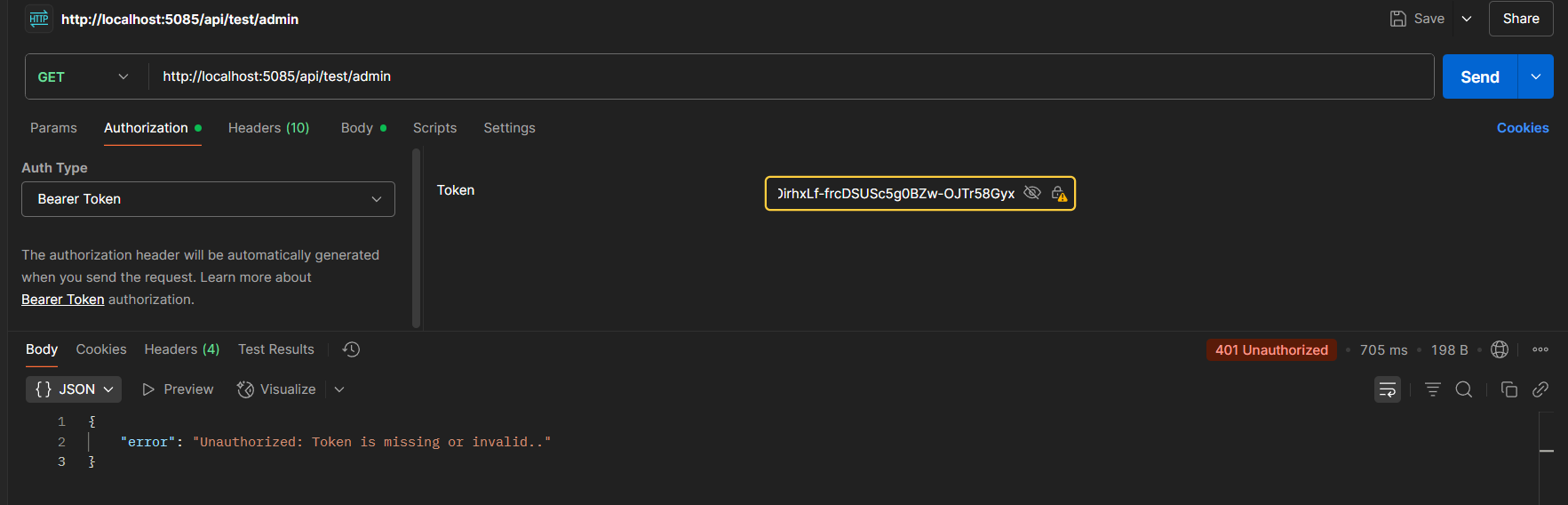
****

**appsettings.json**

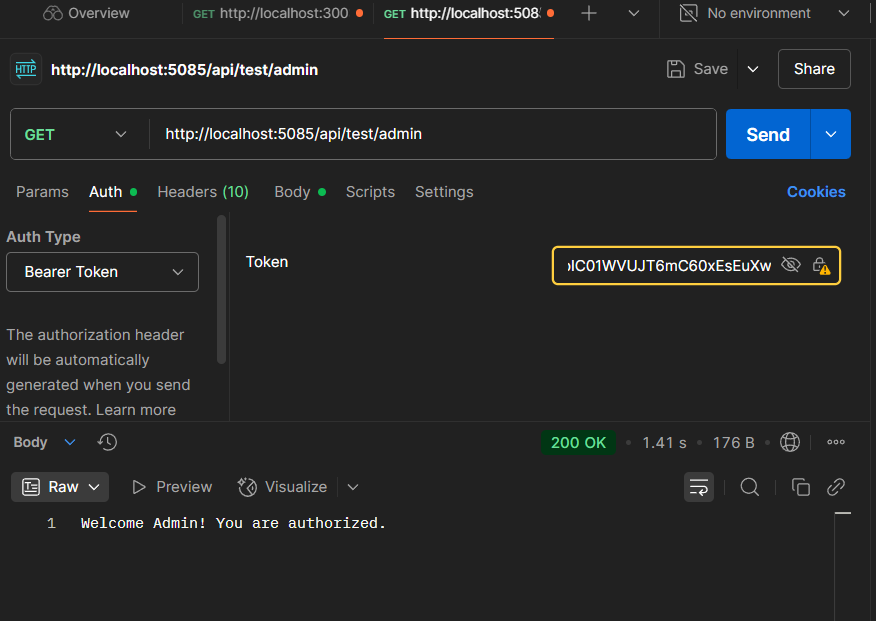
****

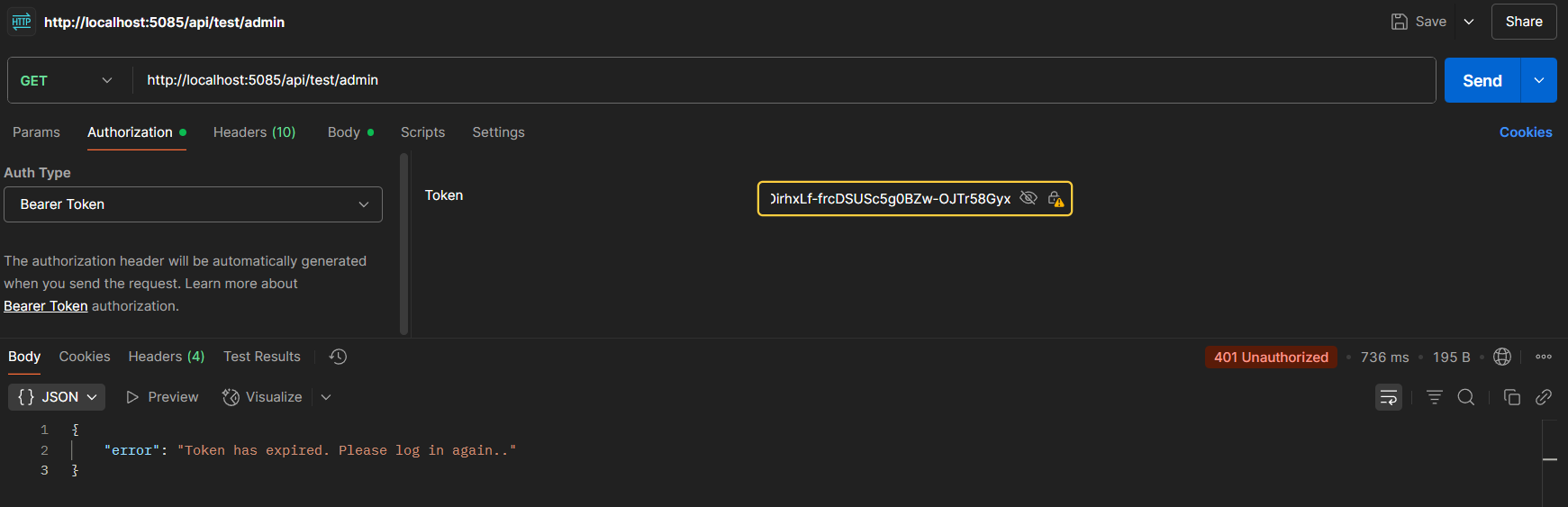
**OUTPUT:**

For Unauthorized User login:

****

Token Before expiring:

****

Token after expiring: ****