Cognizant Nurture 4.0 - Week 5 Hands-on(SupersetID 6363586)

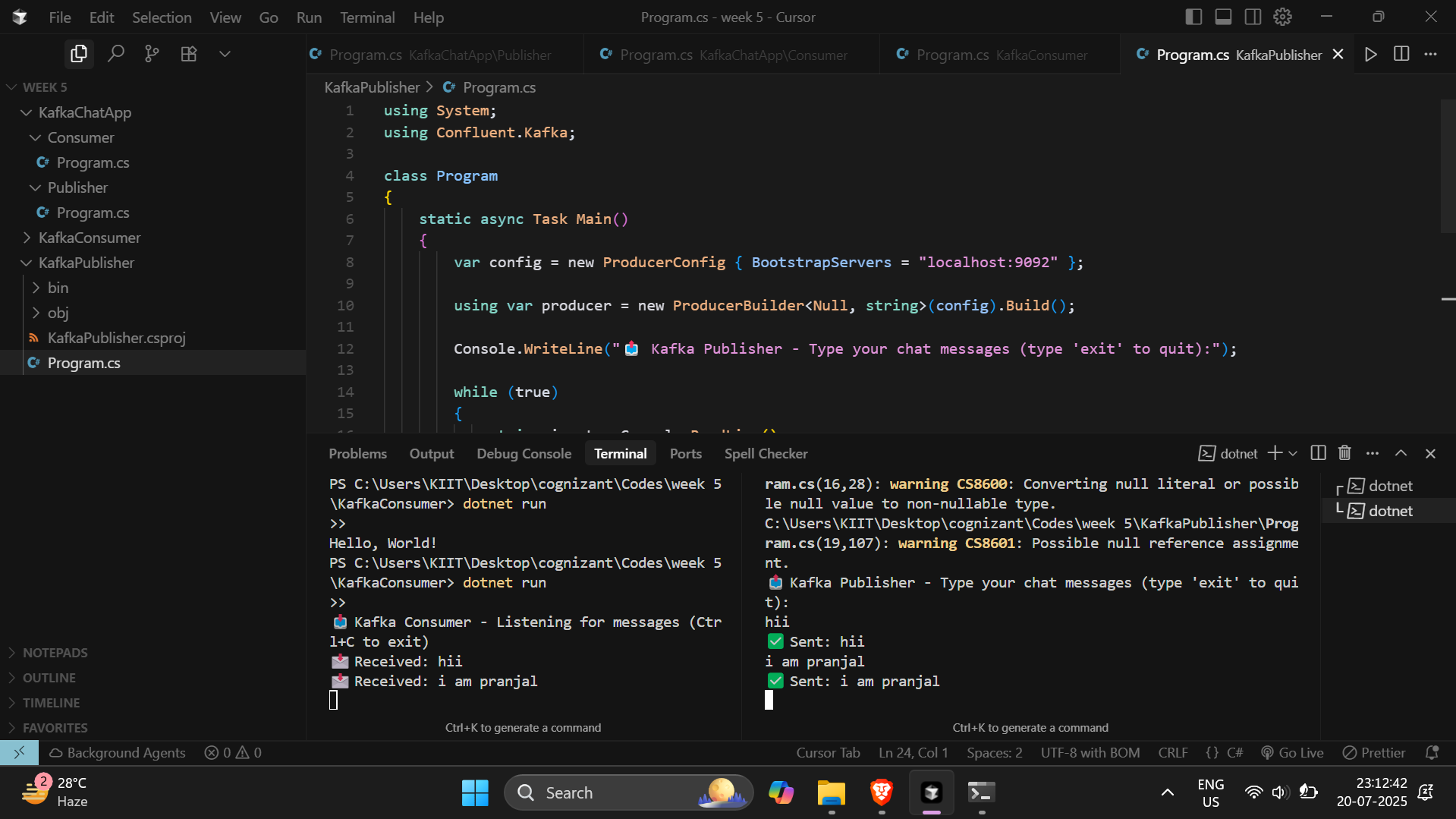
# KafkaPublisher - Program.cs

using System;  
using Confluent.Kafka;  
  
class Program  
{  
 static async Task Main()  
 {  
 var config = new ProducerConfig { BootstrapServers = "localhost:9092" };  
  
 using var producer = new ProducerBuilder<Null, string>(config).Build();  
  
 Console.WriteLine("📤 Kafka Publisher - Type your chat messages (type 'exit' to quit):");  
  
 while (true)  
 {  
 string input = Console.ReadLine();  
 if (input == "exit") break;  
  
 var result = await producer.ProduceAsync("chat-messages", new Message<Null, string> { Value = input });  
 Console.WriteLine($"✅ Sent: {input}");  
 }  
 }  
}

# KafkaConsumer - Program.cs

using System;  
using Confluent.Kafka;  
  
class Program  
{  
 static void Main()  
 {  
 var config = new ConsumerConfig  
 {  
 GroupId = "chat-group",  
 BootstrapServers = "localhost:9092",  
 AutoOffsetReset = AutoOffsetReset.Earliest  
 };  
  
 using var consumer = new ConsumerBuilder<Ignore, string>(config).Build();  
 consumer.Subscribe("chat-messages");  
  
 Console.WriteLine("📥 Kafka Consumer - Listening for messages (Ctrl+C to exit)");  
  
 try  
 {  
 while (true)  
 {  
 var msg = consumer.Consume();  
 Console.WriteLine($"📩 Received: {msg.Message.Value}");  
 }  
 }  
 catch (OperationCanceledException)  
 {  
 consumer.Close();  
 }  
 }  
}

# Kafka Chat App - Execution Screenshot



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

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

**Program.cs**

using Microsoft.AspNetCore.Authentication.JwtBearer;

using Microsoft.IdentityModel.Tokens;

using System.Text;

var builder = WebApplication.CreateBuilder(args);

// JWT Settings from appsettings.json

builder.Services.AddAuthentication(JwtBearerDefaults.AuthenticationScheme)

.AddJwtBearer(options =>

{

options.TokenValidationParameters = new TokenValidationParameters

{

ValidateIssuer = true,

ValidateAudience = true,

ValidateLifetime = true,

ValidateIssuerSigningKey = true,

ValidIssuer = builder.Configuration["Jwt:Issuer"],

ValidAudience = builder.Configuration["Jwt:Audience"],

IssuerSigningKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(builder.Configuration["Jwt:Key"]))

};

});

builder.Services.AddAuthorization();

builder.Services.AddControllers();

// 👇 Set to run on port 5096

builder.WebHost.UseUrls("http://localhost:5096");

var app = builder.Build();

app.UseAuthentication();

app.UseAuthorization();

app.MapControllers();

app.Run();

**Screenshots**

