Web API – Handson\_6

**Chat application in command prompt:**

**Consumer.cs:**

using System;

using Confluent.Kafka;

namespace KafkaChat

{

    public class Consumer

    {

        public static void RunConsumer()

        {

            var config = new ConsumerConfig

            {

                BootstrapServers = "localhost:9092",

                GroupId = "chat-consumer-group",

                AutoOffsetReset = AutoOffsetReset.Earliest

            };

            using var consumer = new ConsumerBuilder<Ignore, string>(config).Build();

            consumer.Subscribe("chat-topic");

            Console.WriteLine("Listening for messages...");

            while (true)

            {

                var consumeResult = consumer.Consume();

                Console.WriteLine($"Received: {consumeResult.Message.Value}");

            }

        }

    }

}

**Producer.cs:**

using System;

using Confluent.Kafka;

namespace KafkaChat

{

    public class Producer

    {

        public static void RunProducer()

        {

            var config = new ProducerConfig

            {

                BootstrapServers = "localhost:9092"

            };

            using var producer = new ProducerBuilder<Null, string>(config).Build();

            Console.WriteLine("Enter messages to send to Kafka (type 'exit' to quit):");

            while (true)

            {

                var message = Console.ReadLine();

                if (message.ToLower() == "exit")

                    break;

                producer.Produce("chat-topic", new Message<Null, string> { Value = message });

                Console.WriteLine($"Sent: {message}");

            }

        }

    }

}

**Program.cs:**

using System;

namespace KafkaChat

{

    class Program

    {

        static void Main(string[] args)

        {

            Console.WriteLine("Type 'p' for Producer or 'c' for Consumer:");

            var choice = Console.ReadLine();

            if (choice == "p")

            {

                Producer.RunProducer();

            }

            else if (choice == "c")

            {

                Consumer.RunConsumer();

            }

            else

            {

                Console.WriteLine("Invalid input");

            }

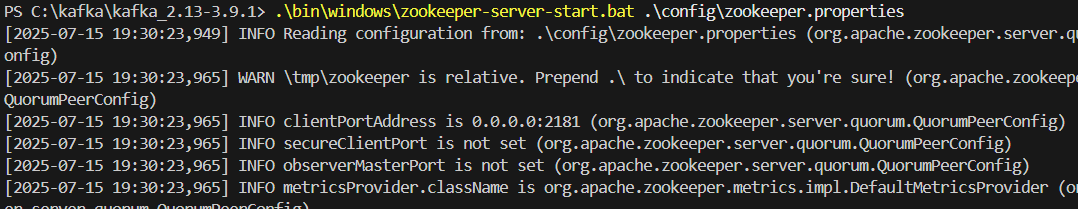
        }

    }

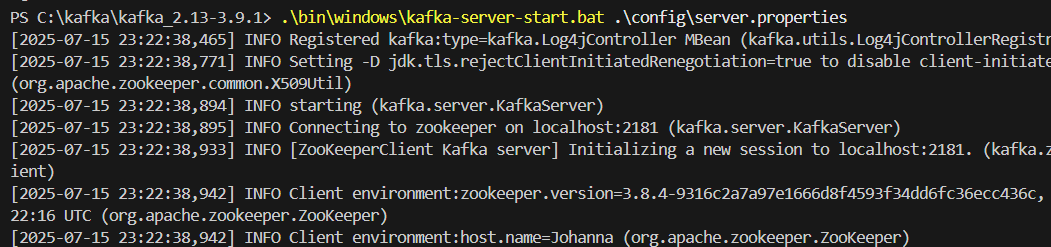
}

**OUTPUT:**

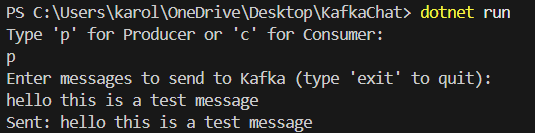
Running zookeeper:

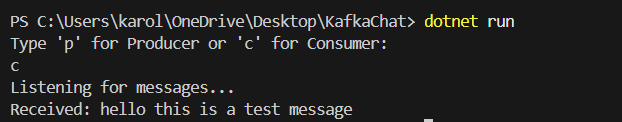


Running Kafka server:



Chat application (Producer/Consumer):





**Chat Application with WinForm:**

**ConsumerForm.cs:**

using System;

using System.Threading;

using System.Threading.Tasks;

using System.Windows.Forms;

using Confluent.Kafka;

namespace KafkaChatManual

{

    public class ConsumerForm : Form

    {

        private ListBox listBoxMessages;

        private CancellationTokenSource cts;

        public ConsumerForm()

        {

            this.Text = "Kafka Chat - Consumer";

            this.Width = 400;

            this.Height = 400;

            listBoxMessages = new ListBox();

            listBoxMessages.Top = 20;

            listBoxMessages.Left = 20;

            listBoxMessages.Width = 340;

            listBoxMessages.Height = 300;

            this.Controls.Add(listBoxMessages);

            this.FormClosing += ConsumerForm\_FormClosing;

            cts = new CancellationTokenSource();

            Task.Run(() => ConsumeMessages(cts.Token));

        }

        private async Task ConsumeMessages(CancellationToken token)

        {

            var config = new ConsumerConfig

            {

                BootstrapServers = "localhost:9092",

                GroupId = "chat-group",

                AutoOffsetReset = AutoOffsetReset.Earliest

            };

            using var consumer = new ConsumerBuilder<Ignore, string>(config).Build();

            consumer.Subscribe("chat-topic");

            try

            {

                while (!token.IsCancellationRequested)

                {

                    try

                    {

                        var consumeResult = consumer.Consume(token);

                        if (consumeResult != null && !string.IsNullOrWhiteSpace(consumeResult.Message.Value))

                        {

                            AddMessageToListBox(consumeResult.Message.Value);

                        }

                    }

                    catch (ConsumeException e)

                    {

                        Console.WriteLine($"Consume error: {e.Error.Reason}");

                    }

                }

            }

            catch (OperationCanceledException)

            {

           //nothing here

            }

            finally

            {

                consumer.Close();

            }

        }

        private void AddMessageToListBox(string message)

        {

            if (listBoxMessages.InvokeRequired)

            {

                listBoxMessages.Invoke(new Action<string>(AddMessageToListBox), message);

            }

            else

            {

                listBoxMessages.Items.Add(message);

            }

        }

        private void ConsumerForm\_FormClosing(object sender, FormClosingEventArgs e)

        {

            cts.Cancel();

        }

    }

}

**ChatForm.cs:**

using System;

using System.Windows.Forms;

using Confluent.Kafka;

namespace KafkaChatManual

{

public class ChatForm : Form

{

private Label lblPrompt;

private TextBox txtMessage;

private Button btnSend;

private Button btnCancel;

public ChatForm()

{

this.Text = "Kafka Chat";

this.Width = 400;

this.Height = 200;

lblPrompt = new Label();

lblPrompt.Text = "Please enter your message here:";

lblPrompt.Top = 20;

lblPrompt.Left = 20;

lblPrompt.Width = 300;

this.Controls.Add(lblPrompt);

txtMessage = new TextBox();

txtMessage.Top = 50;

txtMessage.Left = 20;

txtMessage.Width = 340;

txtMessage.Height = 50;

txtMessage.Multiline = true;

this.Controls.Add(txtMessage);

btnSend = new Button();

btnSend.Text = "Send";

btnSend.Top = 110;

btnSend.Left = 200;

btnSend.Click += BtnSend\_Click;

this.Controls.Add(btnSend);

btnCancel = new Button();

btnCancel.Text = "Cancel";

btnCancel.Top = 110;

btnCancel.Left = 280;

btnCancel.Click += BtnCancel\_Click;

this.Controls.Add(btnCancel);

}

private async void BtnSend\_Click(object sender, EventArgs e)

{

var config = new ProducerConfig { BootstrapServers = "localhost:9092" };

using var producer = new ProducerBuilder<Null, string>(config).Build();

var message = txtMessage.Text;

if (!string.IsNullOrWhiteSpace(message))

{

await producer.ProduceAsync(

"chat-topic",

new Message<Null, string> { Value = message }

);

MessageBox.Show("Message Sent!");

txtMessage.Clear();

}

}

private void BtnCancel\_Click(object sender, EventArgs e)

{

Application.Exit();

}

}

}

**Program.cs:**

using System;

using System.Windows.Forms;

namespace KafkaChatManual

{

static class Program

{

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new ConsumerForm()); //or Application.Run(new ChatForm());

}

}

}

**OUTPUT:**

