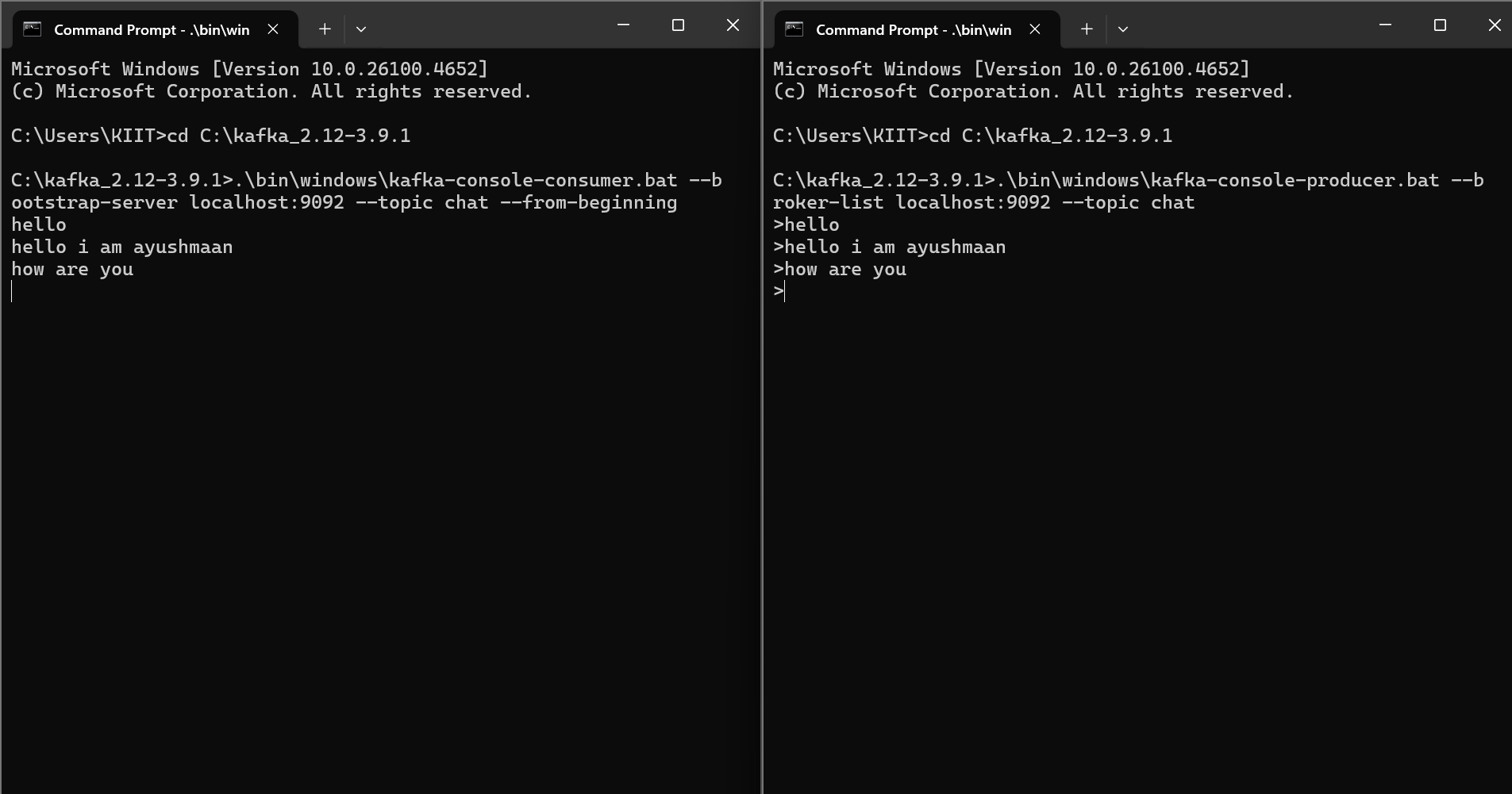
COGNIZANT DN - 4.0 DEEP SKILLING

HANDS ON WEEK-5

ASP.NET Core 8.0 Web API

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



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

PRODUCER CODE:

using Confluent.Kafka;

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();

try

{

var result = await producer.ProduceAsync("chat", new Message<Null, string>

{

Value = txtMessage.Text

});

lblStatus.Text = $"Sent: {txtMessage.Text}";

txtMessage.Clear();

}

catch (Exception ex)

{

lblStatus.Text = $"Error: {ex.Message}";

}

}

CONSUMER CODE:

using Confluent.Kafka;

using System.Reflection;

using System.Threading;

using System.Threading.Tasks;

private CancellationTokenSource cts;

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

{

cts = new CancellationTokenSource();

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

}

private void StartConsumer(CancellationToken token)

{

var config = new ConsumerConfig

{

BootstrapServers = "localhost:9092",

GroupId = "chat-client-group",

AutoOffsetReset = AutoOffsetReset.Earliest

};

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

consumer.Subscribe("chat");

try

{

while (!token.IsCancellationRequested)

{

var cr = consumer.Consume(token);

Invoke((MethodInvoker)(() => lstMessages.Items.Add(cr.Message.Value)));

}

}

catch (OperationCanceledException)

{

consumer.Close();

}

}

protected override void OnFormClosing(FormClosingEventArgs e)

{

cts?.Cancel();

base.OnFormClosing(e);

}

