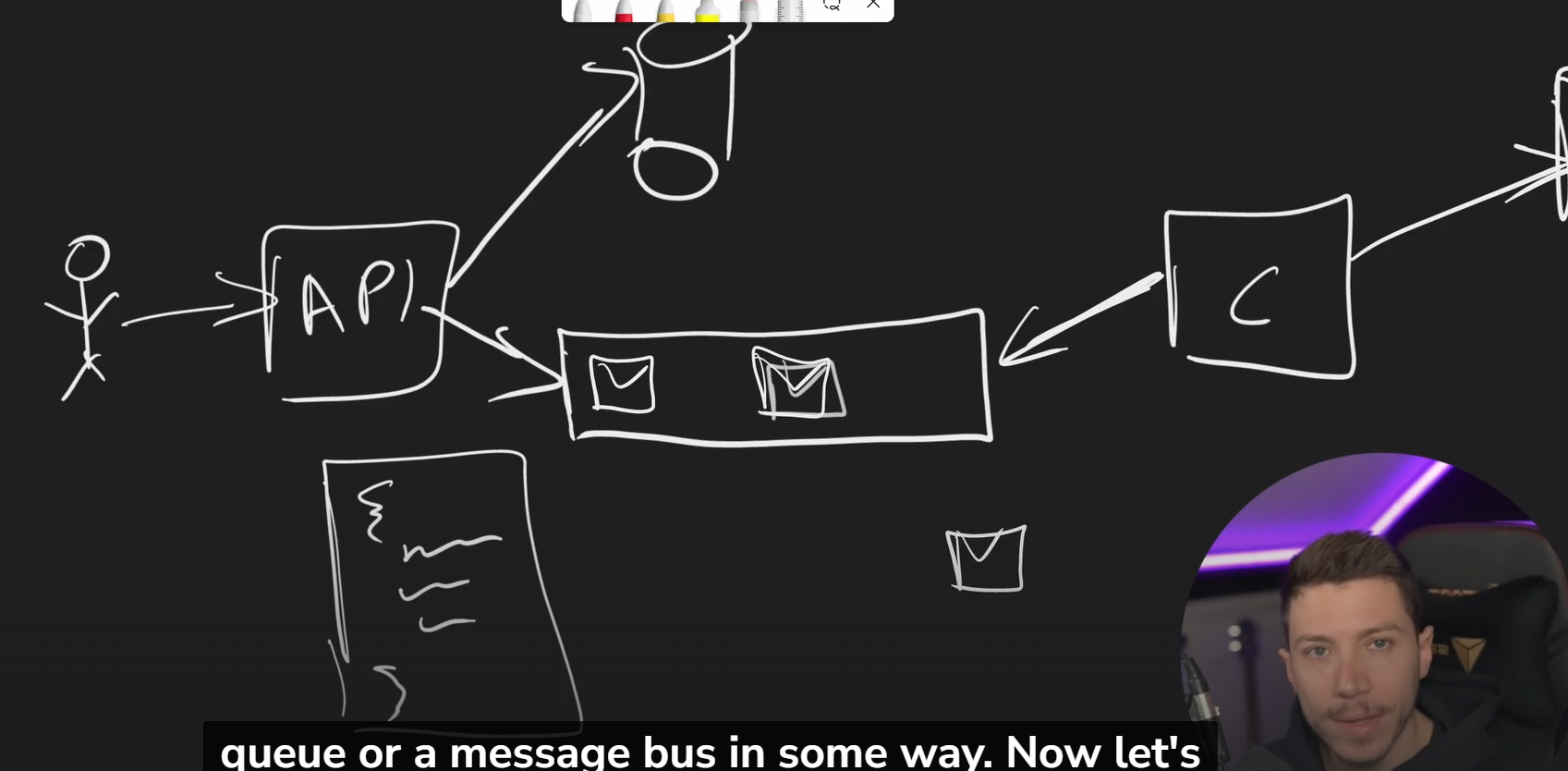
# Cloud Fundamentals: AWS Services for C# Developers

## Course overview

## Getting started with AWS

## AWS SQS

### 3.1. What is a queue?



### 3.2.   Creating our first Queue in SQS

### 3.3.   Publishing through the UI

### 3.4.  Creating a message publisher

using Amazon.SQS;

using Amazon.SQS.Model;

using System.Text.Json;

var sqsClient = new AmazonSQSClient();

var customer = new CustomerCreated {

Id = Guid.NewGuid(),

Email = "mehmetyagci53@gmail.com",

FullName = "Mehmet Yağcı",

DateOfBirth = new DateTime(1982, 1, 1),

GitHubUsername = "mehmetyagci",

};

var queueUrlResponse = await sqsClient.GetQueueUrlAsync("customers");

var sendMessageRequest = new SendMessageRequest {

QueueUrl = queueUrlResponse.QueueUrl,

MessageBody = JsonSerializer.Serialize(customer),

MessageAttributes = new Dictionary<string, MessageAttributeValue> {

{

"MessageType", new MessageAttributeValue

{

DataType = "String",

StringValue = nameof(CustomerCreated)

}

}

}

};

var response = await sqsClient.SendMessageAsync(sendMessageRequest);

Console.WriteLine($"{response.MessageId} {response.HttpStatusCode} {response.ResponseMetadata}");

### 3.5.  Creating a message consumer

using Amazon.SQS;

using Amazon.SQS.Model;

var cts = new CancellationTokenSource();

var sqsClient = new AmazonSQSClient();

var queueUrlResponse = await sqsClient.GetQueueUrlAsync("customers");

var receiveMessageRequest = new ReceiveMessageRequest {

QueueUrl = queueUrlResponse.QueueUrl,

AttributeNames= new List<string> { "All"}, // adding Attributes to the message

MessageAttributeNames = new List<string> { "All" } // adding MessageAttributes to the message

};

while (!cts.IsCancellationRequested) {

var response = await sqsClient.ReceiveMessageAsync(receiveMessageRequest, cts.Token);

foreach (var message in response.Messages) {

Console.WriteLine($"Message Id: {message.MessageId}");

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

// Deleting messages

var deleteMessageResponse = await sqsClient.DeleteMessageAsync(queueUrlResponse.QueueUrl, message.ReceiptHandle);

}

await Task.Delay(1000);

}

### 3.6.  The API we will be working on

### 3.7.   Adding message publishing