MINIO & RabbitMQ

What is Minio –

**MinIO** is a cloud storage server compatible with Amazon S3, released under Apache License v2. As an object store, **MinIO** can store unstructured data such as photos, videos, log files, backups and container images. Foe more details please go through the below link-

<https://docs.min.io/docs/minio-quickstart-guide.html>

What is RabbitMQ –

**RabbitMQ** is an open source multi-protocol messaging broker. Running **rabbitmq**-**server** starts a **RabbitMQ** node in the foreground. The node will display a startup banner and report when startup is complete. To shut down the **server**, use service management tools or rabbitmqctl. For more details please go through the below link-

<https://www.rabbitmq.com/rabbitmq-server.8.html#:~:text=DESCRIPTION,tools%20or%20rabbitmqctl(8).->

Upload File With MINIO Server

Step 1: Declare the variable as per below-

var endpoint = "play.min.io";

var accessKey = "Q3AM3UQ867SPQQA43P2F";

var secretKey = "zuf+tfteSlswRu7BJ86wekitnifILbZam1KYY3TG";

Step 2: Create Connection class and call minio server as per below-

ConnectionFactory connectionFactory = new ConnectionFactory

{

HostName = HostName,

UserName = UserName,

Password = Password,

};

Step 3: Use try and catch block for calling and executing the upload method. The below Run(minio).wait(); method will execute the upload logic.

try

{

var minio = new MinioClient(endpoint, accessKey, secretKey).WithSSL();

Program.Run(minio).Wait();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Step 4: This is the upload logic. File will be uploaded from Outgoing folder to Minio server “vivek” bucket.

private async static Task Run(MinioClient minio)

{

var bucketName = "vivek"; //Bucket name of minio server

var location = "us-east-1"; // Location of minio server

var objectName = "example.pdf"; // file name to be upload

var filePath = "C:\\Users\\user\\.p2\\Desktop\\Outgoing\\example.pdf";

var contentType = "application/pdf"; //Type of file to be upload

Console.WriteLine("Step:1 -- We have example.pdf file in Outgoing folder");

try

{

// Make a bucket on the server, if not already present.

bool found = await minio.BucketExistsAsync(bucketName);

if (!found)

{

Console.WriteLine("Step:2 -- Folder is not found, So we created the folder for you !");

await minio.MakeBucketAsync(bucketName, location); //Making bucket

}

Console.WriteLine("Step:3 -- Upload a file to bucket");

await minio.PutObjectAsync(bucketName, objectName, filePath, contentType);

Console.WriteLine("Successfully uploaded " + objectName);

var uploadPath = "http://play.min.io/minio/vivek/";

Console.WriteLine("Your file saved at location :-"+ uploadPath);

}

catch (MinioException e)

{

Console.WriteLine("File Upload Error: {0}", e.Message);

}

}

Step 5: If you want to save the file from minio server to Incoming folder. We have Run(minio).wait(); method to download the file.

Step 6: Download Logic

public async static Task Run(MinioClient minio)

{

var bucketName = "vivek";

var objectName = "example.pdf";

var filePath = "C:\\Users\\user\\.p2\\Desktop\\Incoming\\" + objectName;

Console.WriteLine("Step:4 -- We have example.pdf file at Minio Server in vivek folder");

try

{

await minio.GetObjectAsync(bucketName, objectName,filePath);

Console.WriteLine("Step:5 -- File Downloaded Successfully, path is- "+ "C:\\Users\\user\\.p2\\Desktop\\Incoming");

}

catch (Exception ex)

{

Console.WriteLine("Unable to Download : " + ex.ToString());

}

Console.WriteLine("Press Enter to Send link via message to RabbitMQ ");

Console.ReadLine();

}

Step 6: Now we are going to use RabbitMQ server as a localhost.

(a) : Download the Erlang first with link- <https://www.erlang.org/downloads>

(b) : Now download RabbitMQ server from link-

<https://www.rabbitmq.com/download.html>

(c) : Install RabbitMQ management plugin

By default, the RabbitMQ Windows installer registers RabbitMQ as a Windows service, so technically we’re all ready to go.  In addition to the command line utilities provided for managing and monitoring our RabbitMQ instance, a Web-based management plugin is also provided with the standard Windows distribution.  The following steps detail how to get the management plugin up and going.

First, from an elevated command prompt, change directory to the sbin folder within the RabbitMQ Server installation directory (e.g. %PROGRAMFILES%\RabbitMQ Server\rabbitmq\_server\_2.7.1\sbin).

Next, run the following command to enable the rabbitmq management plugin:

rabbitmq-plugins.bat enable rabbitmq\_management

Lastly, to enable the management plugin we need to reinstall the RabbitMQ service.  Execute the following sequence of commands to reinstall the service:

rabbitmq-service.bat stop

rabbitmq-service.bat install

rabbitmq-service.bat start

(d) : Now just run the Rabbit MQ from browser as a localhost with url-

<http://localhost:15672/>

username- guest (bydefault)

password- guest (bydefault)

Step 7: Let’s right the logic to connect with RabbitMQ and send message as a producer.

public static void Rabbit\_Send()

{

var factory = new ConnectionFactory() { HostName = "localhost" }; //create connection with rabbitmQ server

using (var connection = factory.CreateConnection())

using (var channel = connection.CreateModel()) //Create channel

{

channel.QueueDeclare(queue: "Minio",

durable: false,

exclusive: false,

autoDelete: false,

arguments: null);

string message = "http://play.min.io/minio/vivek";

var body = Encoding.UTF8.GetBytes(message); //Encode the body of message

channel.BasicPublish(exchange: "",

routingKey: "Minio",

basicProperties: null,

body: body); //publish the message

Console.WriteLine("Message has been sent ! Please check your Inbox");

}

Console.WriteLine(" Press Enter to Consume the message ");

Console.ReadLine();

}

Step 8: Open RabbitMQ server, go to the Queue option find your generated queue name in my case it is Minio. Now, click on this and you can get your message by GetMessage option.

Step 9: Now, let’s right the logic to consume the message as consumer.

public static void Rabbit\_Received()

{

var factory = new ConnectionFactory() { HostName = "localhost" };

using (var connection = factory.CreateConnection())

using (var channel = connection.CreateModel())

{

channel.QueueDeclare(queue: "Minio", durable: false, exclusive: false, autoDelete: false, arguments: null);

var consumer = new EventingBasicConsumer(channel);

consumer.Received += (model, ea) =>

{

var body = ea.Body;

var message = Encoding.UTF8.GetString(body.Span); //use Span for compatibility else it will throw error.

Console.WriteLine("Messgaed received, Here-- ", message);

};

channel.BasicConsume(queue: "Minio", autoAck: true, consumer: consumer);

Console.ReadLine();

}

Step 10: Now let’s combine all the code and method into Program class and call from Program.cs class one by one to execute the full logic.

Step 11: Here program class goes as per below-

using MiniIO\_RabbitMQ;

using Minio;

using Minio.Exceptions;

using RabbitMQ.Client;

using System;

using System.Threading.Tasks;

public class Program : Send

{

private const string UserName = "guest";

private const string Password = "guest";

private const string HostName = "localhost";

static void Main(string[] args)

{

ConnectionFactory connectionFactory = new ConnectionFactory

{HostName = HostName, UserName = UserName,Password = Password,};

Send sd = new Send();

var endpoint = "play.min.io";

var accessKey = "Q3AM3UQ867SPQQA43P2F";

var secretKey = "zuf+tfteSlswRu7BJ86wekitnifILbZam1KYY3TG";

try

{

var minio = new MinioClient(endpoint, accessKey, secretKey).WithSSL();

Program.Run(minio).Wait();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

//call Download class to download the file from minio server

string s = "yes";

Console.WriteLine("You want to save you file in local system?"+ "Please enter "+ s +" for download the file");

string ss=Console.ReadLine();

if (s == ss)

{

var minio = new MinioClient(endpoint, accessKey, secretKey).WithSSL();

Download.Run(minio).Wait();

}

//call Rabbitmq send class to send the message as a producer to rabbitmq localhost

Console.WriteLine("Calling Rabbit\_Send Method to send the message as a Producer ! ");

var sx=Send.Rabbit\_Send();

Console.WriteLine("Sending operation is : "+ sx);

//call Rabbitmq received class to received the message as a consumer from rabbitmq localhost

Console.WriteLine("Calling Rabbit\_Received Method to received the message as a Consumer !");

Received.Rabbit\_Received();

Console.ReadLine();

}

// File uploader task.

private async static Task Run(MinioClient minio)

{

var bucketName = "vivek";

var location = "us-east-1";

var objectName = "example.pdf";

var filePath = "C:\\Users\\user\\.p2\\Desktop\\Outgoing\\example.pdf";

var contentType = "application/pdf";

Console.WriteLine("Step:1 -- We have example.pdf file in Outgoing folder");

try

{

// Make a bucket on the server, if not already present.

bool found = await minio.BucketExistsAsync(bucketName);

if (!found)

{

Console.WriteLine("Step:2 -- Folder is not found, So we created the folder for you !");

await minio.MakeBucketAsync(bucketName, location);

}

Console.WriteLine("Step:3 -- Upload a file to bucket");

await minio.PutObjectAsync(bucketName, objectName, filePath, contentType);

Console.WriteLine("Successfully uploaded " + objectName);

var uploadPath = "http://play.min.io/minio/vivek/";

Console.WriteLine("Your file saved at location :-"+ uploadPath);

}

catch (MinioException e)

{

Console.WriteLine("File Upload Error: {0}", e.Message);

}

}

}