

رشته: پردازش ابری

روز دوم

Description of project and tasks

This module is 4 hours - **IoT + AI + Web service**

You are a Cloud Engineer working at Unicorn Company which delivers Enterprise AI service. Recently, your company plan to build a new AI service which aimed to analyze short messages that collecting from IoT devices to detect the emotions hidden behind the messages. Then share the result to public by hosting a web service.

You are assigned as solutions architect for this project; Your responsibility is to design and build the technical solution.

You need to configure IoT client (acting as IoT devices) to be able to publish messages to MQTT broker. You have to build the rest of stuffs on a Public Cloud Platform as per business requirement.

Then you will need to analyze these messages by utilizing AI service and store the result in a NoSQL database. In the end, you need to build a web application to serve the result in NoSQL database to customers.

As a solutions architect, you may be required to perform many different types of tasks. This challenge will gauge your ability to change tasks and respond to ambiguity.

Background

Today's business world is changing with the adoption of IoT (Internet of Things). IoT is helping in prominently capturing a tremendous amount of data from multiple sources. However, wrapping around the multitude of data coming from countless of IoT devices, makes it complex to collect, process, and analyze the data.

Realizing the future and full potential of IoT devices will require an investment in new technologies. The convergence of AI (Artificial Intelligence) and IoT can redefine the way industries, business, and economies functions. Unicorn Company value this project very much.

Out of security concerns, you can use any existing IAM user or role to handle the challenges. But please don't create any other IAM user/role from IAM console during the challenge.

Re-platform schedule

Phase I

- Find a way to send and also receive the messages sending from vendor IoT Client (which acts as IoT devices), you need to consider the AWS IoT Core service here as your IoT Broker and EMQX client app as your MQTT client.

Phase II

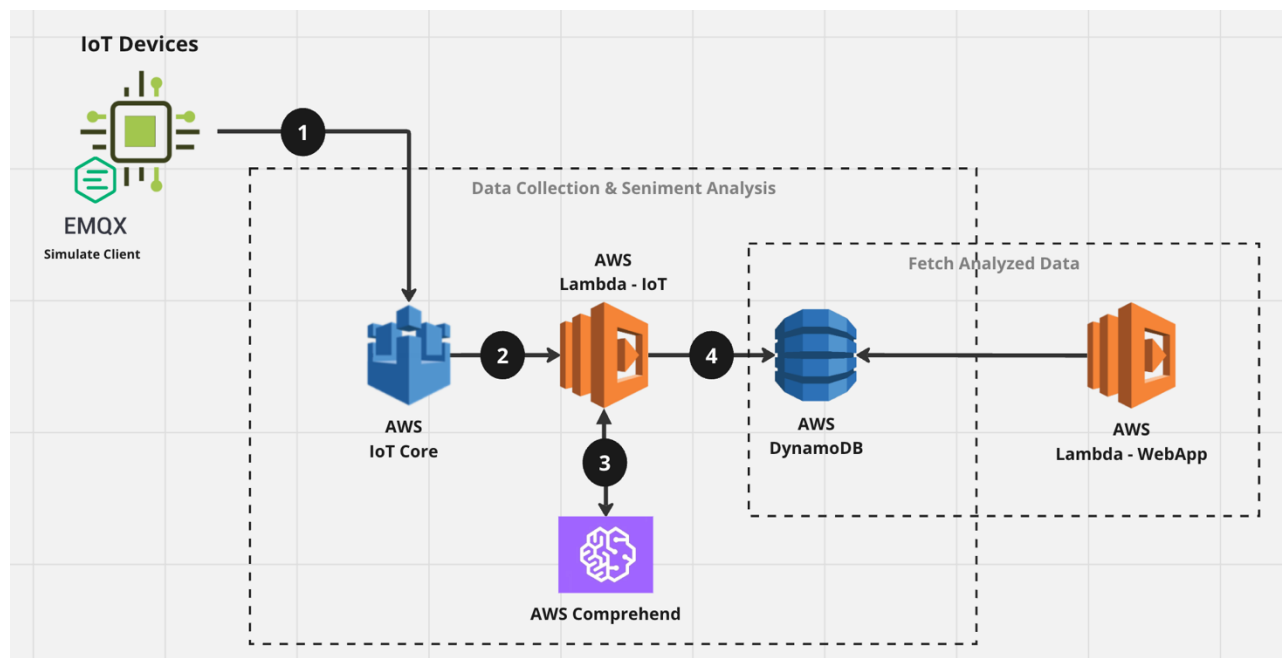
- Detect the sentiment from the messages, store the analytic result to NoSQL database. You may need to consider the following AWS Services: DynamoDB, Lambda, Comprehend.

Phase III

- Serve the result by WebApp, you need to provide http URL in the **document file**.

Technical Details

For this day of competition focus on Lambda as your computing resource. **Do not use EC2, ECS or Fargate or EKS!**



The above example illustrates one simple architectural design for the deployment of the application. This shows some segments needed to serve requests. As an architect of Unicorn company, you have the responsibility to design a modern architecture for Unicorn Service by leveraging powerful cloud services.

Service Details

IoT Core

You need to setup and configure AWS IoT Core service, in the way that it could receive MQTT messages from client (IoT devices). Use **'irsc/skill53'** as your topic name.

IoT Device (IoT Client)

EMQX client application has been already installed on your host machine. This client can be used to help you test our sending different messages and check whether if you are able to receive and process it on the IoT Core service. Alongside this client application, a file containing lots of sample messages also have been located on your desktop (messages.txt), to help you test our your underlying IoT infrastructure.

Lambda - IoT

You have to create a Lambda function that fetches proper IoT device messages and then analyze them through AWS Comprehend and then store the output sentiments to AWS DynamoDB.

AWS Comprehend

Use any regions that AWS Comprehend is supported on to enable analysis the message data for processing sentiments. Make sure that you lambda functions could perfectly work with this AWS service.

Lambda - WebApp

You have to develop a web application to host the web service that serves the Analyzed data which you stored in the DynamoDB.

This web application should have the following structure of request/response:

```
# Sample Request - Get Specific Value
curl -X GET [your_endpoint]/sentiment/[id]

# Web Application Response
'{"id": "[uuid]", "sentiment": "[sentiment]", "message": "[message]"}'
```

```
# Sample Request - Get All sentiments
curl -X GET [your_endpoint]/sentiments

# Web Application Response
'[{ "id": "[uuid]", "sentiment": "[sentiment]", "message": "[message]"}, ...]'
```

Hints of following best practices on Lambda functions:

- You can develop this function in the way that it connects to your AWS account using AWS Config.
- You can use **the existing IAM** role in your account to make your Lambda Function able to connect to DynamoDB without password.

Database

You will need to deploy a DynamoDB database that is highly available and **centralized** to store the IoT analytic result. The table name should be **unicorn**, structure would be like following:

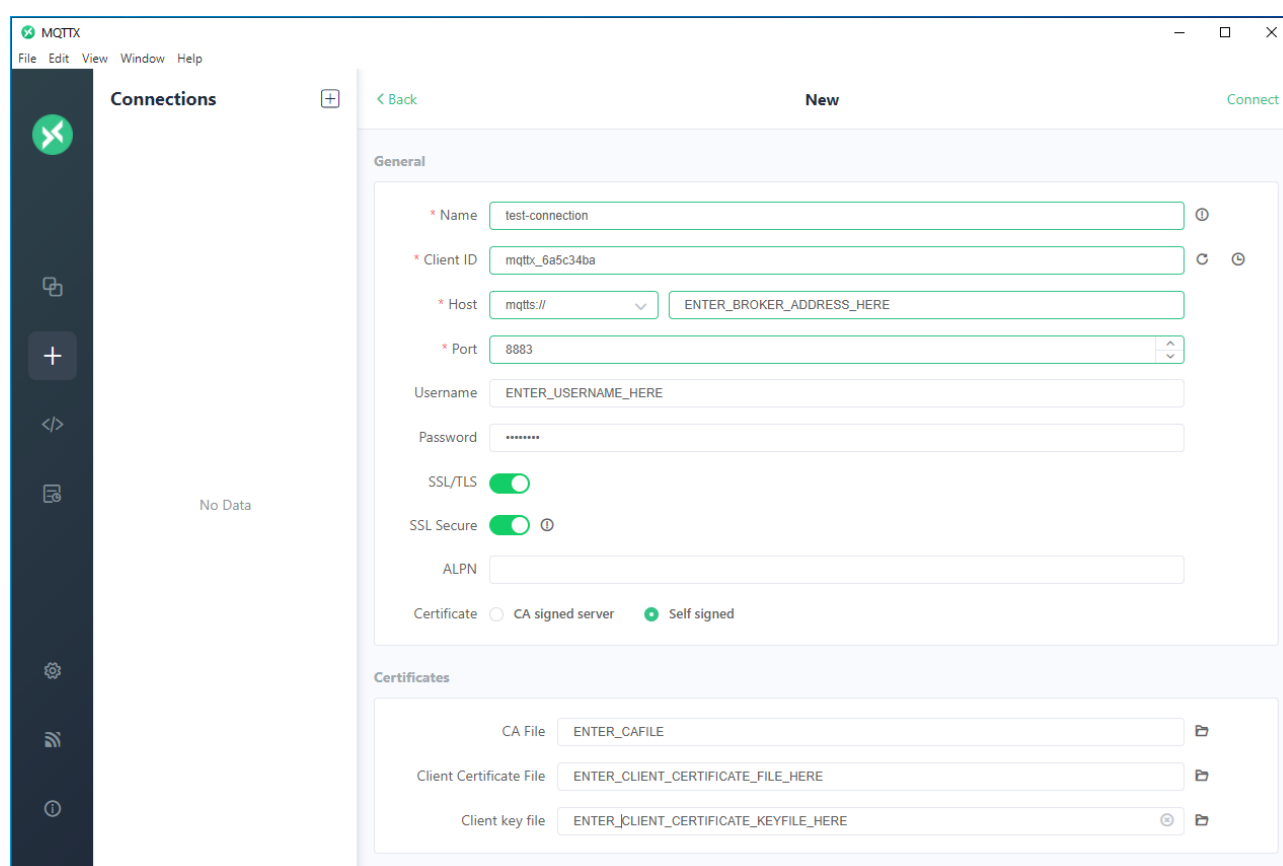
Key_Name	Key_Description	Comment
id	Partition Key	
sentiment	Sort Key	The result after sentiment analysis
message		The messages received from IoT devices

Submit your answer in the document

- **HTTP-URI:** URL for the web service you built
- **IoT-Core-EP:** Your IoT Core endpoint

Hint on Setting-Up IoT Client

1. Open MQTTX app on your desktop
2. Create a New Connection
3. Follow **Name, Host, Port, SSL/TLS, SSL Secure, Certificates** with proper AWS configuration (sample as following figure):



The screenshot shows the MQTTX application interface. On the left is a dark sidebar with icons for connections, settings, and other functions. The main window is titled 'MQTTX' and has a menu bar with 'File', 'Edit', 'View', 'Window', and 'Help'. The 'Connections' panel on the left shows 'No Data'. The 'New' connection configuration window is open, showing the following fields:

- General**
 - * Name: test-connection
 - * Client ID: mqttx_6a5c34ba
 - * Host: mqtt:// (dropdown) ENTER_BROKER_ADDRESS_HERE
 - * Port: 8883
 - Username: ENTER_USERNAME_HERE
 - Password: *****
 - SSL/TLS: ☒
 - SSL Secure: ☒ ⓘ
 - ALPN:
 - Certificate: ☐ CA signed server ☒ Self signed
- Certificates**
 - CA File: ENTER_CAFILE
 - Client Certificate File: ENTER_CLIENT_CERTIFICATE_FILE_HERE
 - Client key file: ENTER_CLIENT_CERTIFICATE_KEYFILE_HERE