Balai Pengembangan Talenta Indonesia
Pusat Prestasi Nasional
Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi

MERDEKA BELAJAR

LKS SMK TINGKAT NASIONAL

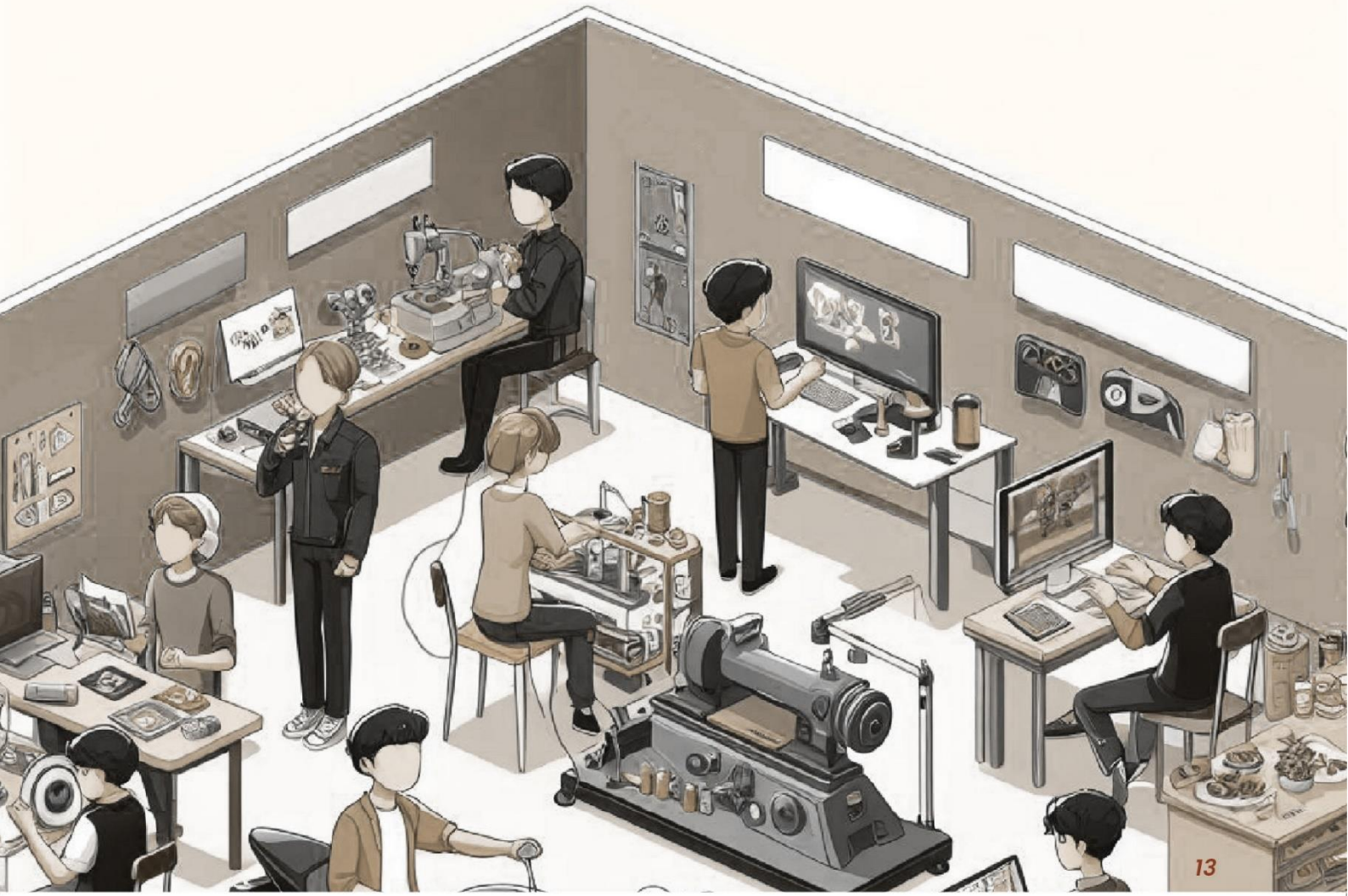
SMK

Soal

Lomba Kompetensi Siswa Nasional 2024

Komputasi Awan

(Cloud Computing)



Description of project and task – Day 1

This module is 4 hours.

You are Cloud Engineer working at a Start-up Company. Recently, your company planned to build a new service to analyse short messages collected from IoT (Internet of Things) devices. Then share the result to the public by hosting a web service.

You are assigned as a solution architect for this project; Your responsibility is to design and build the technical solution. A partner company manages the Internet of Things (IoT) devices, so you do not need to worry about the configuration of these devices. But you must build the rest of the stuff on a Public Cloud Platform as per business requirements.

The IoT devices will keep sending messages to you from the event's start time. You will get points if you can successfully receive these messages. Then, you must analyse these messages and store the result in a Relational Database Service RDS). You must build a web application to serve customers' results in the RDS and set prepare for Machine Learning.

As a solutions architect, you may be required to perform many unusual 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. IoT is helping to capture a tremendous amount of data from multiple sources. However, wrapping around the multitude of data from countless IoT devices, makes it complex to collect, process, and analyse the data.

Realizing the future and full potential of IoT devices will require an investment in innovative technologies. The IoT convergence can redefine how industries, businesses, and economy's function. Startup Company values this project very much.

According to the feedback from the operation team, using serverless is enough to run Startup Service reliably. But you need to monitor the traffic closely in case of any unexpected traffic and ensure Startup Service is available.

Task 1

1. Understand architectural problems that have already been prepared.
2. Read the documentation thoroughly (Outline below),
3. Log in to the Amazon Web Service Console.
4. Set up the IoT service using CloudFormation. You can read IoT Service details.
5. Configure and run the program. You can read the Publisher details.
6. Set up the IoT rule. You can read IoT Rule services details.
7. Set up the Virtual Private Cloud and Security Group. You can read the Networking details.
8. Set up the Database. You can read the Database service details.
9. Set up the Lambda Function for the process data from IoT Service. You can read Lambda services details.
10. Set up the bucket to collect data from device using CloudFormation. You can read Bucket Data Service details.
11. Set up the data stream. You can read the data stream service details.
12. Set up the API Gateway. You can read the API Gateway service details.
13. Create the data models. You can read the Data Models service details.
14. Verify the apps.

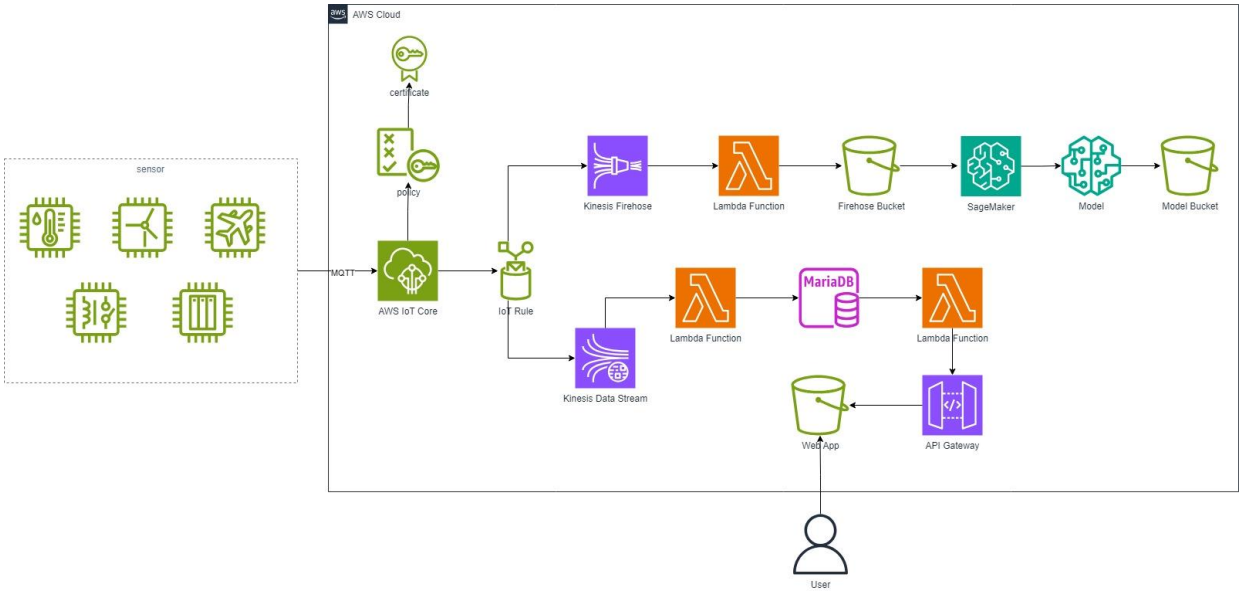
Technical Details

1. The source code is already from GitHub. You can access it from this link: https://github.com/handipradana/lksn2024.git,
2. The default region is **Oregon,**
3. Resource AWS IoT Core, AWS IoT Rule, Networking, Database, S3 Bucket, and Sage Maker Notebook instance must use CloudFormation. You must create CloudFormation using YAML/JSON. If you create manual or use Visual Design, **you lose the point**.

Reference

1. For documentation IoT Services you can access this link: https://docs.aws.amazon.com/iot/index.html
2. For documentation on S3 Bucket Policy Services you can access this link: https://docs.aws.amazon.com/AmazonS3/latest/userguide/example-bucket-policies.html
3. For documentation CloudFormation you can access this link: https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/Welcome.html
4. For documentation SNS you can access this link: https://docs.aws.amazon.com/sns/latest/dg/sns-getting-started.html
5. For documentation AWS API Gateway, you can access on this link: https://docs.aws.amazon.com/apigateway/latest/developerguide/welcome.html
6. For documentation AWS SageMaker, you can access on this link: https://docs.aws.amazon.com/sagemaker/latest/dg/gs.html?icmpid=docs_sagemaker_lp/index.html
7. For documentation AWS Kinesis Data Stream, you can access on this link: https://docs.aws.amazon.com/streams/latest/dev/introduction.html
8. For documentation AWS Kinesis Firehose, you can access on this link: https://docs.aws.amazon.com/firehose/latest/dev/what-is-this-service.html
9. For documentation Python Machine Learning, you can access on this link: https://realpython.com/tutorials/machine-learning/

Architecture

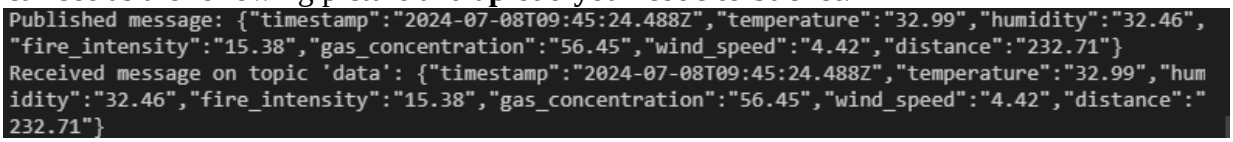


AWS IoT Core

To successfully receive the message from IoT Devices, you need to create the device and policies using CloudFormation if you don't want to lose points, (**don’t use the visual builder, block, or console**), set the name of stack is lksiot, give the thing name is **lksiot**, set the policies, and give the name is lksiotpolices, allow the policies only Connect, Publish, Receive and Subscribe.

Publisher

The publisher in MQTT (Message Queuing Telemetry Transport) is the entity responsible for sending messages to the MQTT broker. Open visual studio code, open file, and modify source code. The source code you can access from GitHub in the iot folder. Before you modify source code, you need library **aws-iot-device-sdk-v2, witson, dotenv**. Create the environment variable and setting the variable ENDPOINT, CLIENT_ID, TOPIC **(datasensor)**, CERT_PATH, KEY_PATH, ROOT_CA_PATH. Run the main program **node main.js**. If you successfully, you can see as the following picture and u**pload your code to bucket**.



IoT Rules

The rules for storing data from the device to Kinesis Data Stream and Firehose, set the 2 rules using CloudFormation, if you don't want to lose points, (**don’t use the visual builder, block, or console**) and give the name:

1. **lks_datastream_yourprovince**, as an example: **lks_datastream_papuabarat**. Setting the query **SQL from your topic** and setting the description as ***Rules for Devices connect to Analytics***.
2. **lks_firehose_province** as an example: **lks_firehose_papuabarat**. Setting the query **SQL from your topic** and setting the description as ***Rules for Devices connect to Kinesis***.

Networking

You must build a network infrastructure before you start the Application. You should create an infrastructure using **CloudFormation,** and set the stack using YAML or JSON**,** give the name the stack is ***lkscf-networking*** and according to the following instructions. Set the name VPC is lks-2024-vpc, CIDR 172.15.0.0/16, have public subnet and private subnet at 3 availability zone from 172.15.0.0/24 until 172.15.5.0/24, have 2 route table, internet gateway and 1 NAT Gateway, and finally you must the security group for RDS (3306) and give the name is SG-DB, allow the SG to all private subnets.

Database

Relational Database Services (RDS) has a rule for the save data from Kinesis DataStream using Lambda Function, create the RDS (MariaDB – Free Tier) using CloudFormation, **if you do not use CloudFormation you lose point**, Give the name stack is **lksdb**. RDS configuration according to the following instructions: set of the name Database is dbiot, set the username admin

and password **Lksncc2024!**, set the db instance class is db.t3.micro, and configuration the VPC is your create VPC, public access is no.

Lambda Function

The Lambda function is used to trigger data from the kinesis data stream and firehose which can later send notifications, save to the database, and call data from the database so that it can be accessed by the website or processed into a model in machine learning. There are 4 lambdas that you must create. Do the configuration according to the following instructions.

1. Lambda Datastream

AWS Lambda function is used to provide a trigger obtained from AWS IoT Core service save data to Relational Databases Service (RDS MariaDB). Set a lambda function with **the name lks-*lambda-store-datastream*** with a runtime using **NodeJs20x** and use the ***labrole***, setting the environment variables, use the variable is DB_HOST, DB_USER, DB_PASSWORD, DB_NAME and DB_TABLE_NAME (tbdatasensor), set the VPC using 3 private subnets, setting the trigger from Kinesis Data Stream and **insert the code.**

2. Lambda GetData

AWS Lambda function is used to provide a trigger obtained from AWS IoT Core service to read data to Relational Databases Service (RDS MariaDB). Set a lambda function with **the name lks-*lambda-getdata-db*** with a runtime using **NodeJs20x** and use the ***labrole***, setting the environment variables, use the variable is DB_HOST, DB_USER, DB_PASSWORD, DB_NAME and DB_TABLE_NAME (tbdatasensor), set the VPC using 3 private subnets, setting the trigger from API Gateway and **insert the code.**

3. Lambda Firehose

AWS Lambda function is used to provide a trigger obtained from Kinesis service to store data to S3 Bucket Firehose. Set a lambda function with **the name lks-*lambda-firehose*** with a runtime using **NodeJs20x** and use the ***labrole***, setting the environment variables, use the variable is BUCKET_NAME, JSON_FILE_NAME, and CVS_FILE_NAME and **insert the code.**

S3 Bucket

Create the bucket using **CloudFormation, if you create manual from AWS Console.** Give the name stack **is lksbucket, you lose the point, please follow the instruction on this bellow:**

- Bucket 1

The bucket is used to website static. Make the bucket name with the name lks-***bucketweb-yourprovince***, for example is lks-***bucketweb-papuabarat*** in the **Oregon Regions.** The permission issue so that **can be accessed by public using the bucket policy.** If you do not understand, please read the documentation about the bucket policy in the Technical Details**.** Set the bucket to static website hosting**.** Upload file index.html and script.js from github.

- Bucket 2

The bucket is used to store data from DataStream. Make the bucket name with the name ***lks-bucketfirehose-yourprovince***, for example is ***lks-bucketfirehose-papuabarat*** in the **Oregon Regions**. The permission issue so that **can be accessed by public using the bucket policy and versioning**.

- Bucket 3

The bucket is used to store data from Sagemaker. Make the bucket name with the name ***bucketmodel-yourprovince***, for example is ***lks-bucketmodel-papuabarat*** in the **Oregon Regions**. The permission issue so that **can be accessed by restricted by your IP public using the bucket policy**. Make archive the file when it has been 30 days and delete a permanent entire file after 1 year.

Data Stream

AWS Kinesis is a service that provides a platform for streaming data in real-time on the AWS cloud. The two main services are AWS Kinesis Data Streams which serves to deliver data as a record and AWS Kinesis Data Firehose which has services to handle scale, monitoring, and infrastructure management, which in turn must deliver data to Amazon S3. Follow configuration on this bellow.

- Kinesis Data Stream
  Configure the data stream, give the name is **lksdatastreamiot** and set the capacity mode on-demand.
- Kinesis Data Firehose
  Firehose is fully managed service for delivering data to S3, following the instructions, set the source the direct PUT and destination to S3 Firehose. Give the name of the Firehose Stream is **LKS-Firehose-2024** and set the transform with AWS Lambda. Set the Firehose Lambda, set the Destination to your bucket firehose and set the time zone is Asia/Jakarta, for the role you can use the **labrole.**

API Gateway

This API gateway is used to bridge between the database and the application, configure the API Gateway and name it **lks-apigateway**, set the endpoint type is regional, the resource use /data, set the method is GET and using lambda **lks-lambda-getdata-db**, set the CORS and set the Default 4xx and 5xx, Access Control Allow Methods for GET. Set the stage give the name is production.

SageMaker

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. Create the notebook instance using **cloudformation** (don’t use manual) give the name of stack is **lksml**, for the instance set the name is iotml-yourprovince, instance type is ml.t3.medium, platform Amazon Linux 2 jupiter Lab 3. Input the existing code on GitHub and complete it according to the following algorithm.

1. Set your name bucket firehose
2. Set your name bucket models
3. Set your name data json
4. Save model to .pkl local file
5. Upload model to S3 Model

Verify the result

If your success for deploy instance, you can see the Dashboard.