DevOps candidate's home exam

As a part of your recruitment process, You need to take the DevOps exam and submit the assignment.

Intro:

In this exam, You need to create a system of 2 Docker microservices in Python (or any other language you are familiar with) built on ECS/EKS and use S3 bucket, Elastic load balancer, and SQS.

Jenkins / Another CI/CD tool.

All creations should be written by IaC (CloudFormation\Terraform\CDK).

You'll get AWS permission to our account for this exam.

Tasks:

1. Create a Jenkins / Another CI/CD tool for making CI/CD processes
2. Build your cloud environment using an IaC tool (Cloudformation/Terraform)
3. Create an SQS and an S3 bucket
4. Microservice 1
   a. A rest microservice that gets requests from an ELB and sends them to the service
      The service should listen to a port and get the requests
      Request payload example:

```
{
  "data": {
    "email_subject" : "Happy new year!",
    "email_sender" : "John doe",
    "email_timestream": "1693561101",
    "email_content": "Just want to say... Happy new year!!!"
  },
  "token" : "$DJISA<$#45ex3RtYr"
}
```

   The data of the payload should be published to the SQS after a process of token validation. The token should be stored in SSM parameter store or other secrets manager for your choice in your code for comparison.

   The task should validate:

- The token correctness
- Make sure the date is valid , e.g has the 4 text fields.

5. Microservice 2
   a. An SQS microservice that pulls the SQS messages, and upload them into the S3 that created before (You can choose the path)
   b. The service should pull every X time and check if there is a waiting message to upload into the S3
6. CI jobs - Create a CI job for both services, The CI should build the Docker image and then push it to a docker repo (Dockerhub/ECR/etc)
7. CD jobs - Create a CD job for both services, The CD should get the version of the image and deploy it on the environment you created using the IaC

Bonus:

1. Create tests for the process
2. Add some monitor tool for the CI/CD process and the microservices activity (Grafana or Prometheus or similar tool)

Important

Your work should be executable and explained for the ability to test it in our side.
Please mention in a README what are the steps for running the code

Highlights:

1. All code should be pushed to a Git repo, The repo should be public so we can check on it, When you submit the assignment add the repo link
2. README - Add a doc of README for make the read process more comfortable
3. All above AWS resources can be created on AWS instead of Localstack (Your personal AWS account) using a free tire instances only.