Platform Engineer – Technical Assessment

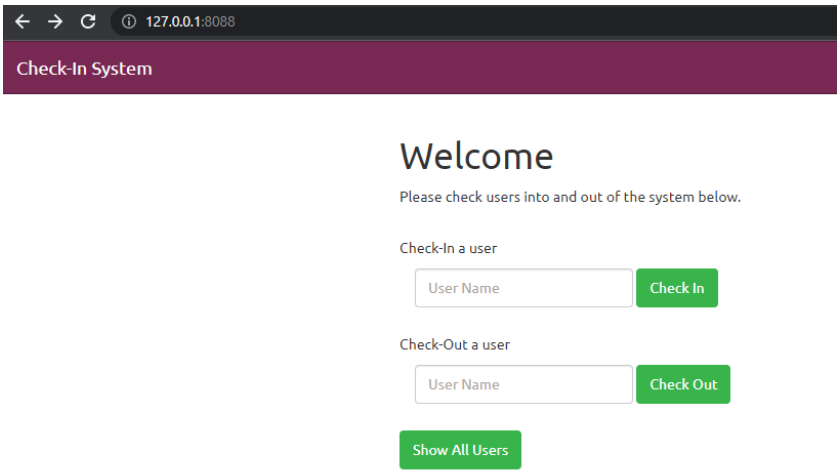
STAGING

Prerequisites

1. Install Python
2. Install Visual Studio Code
    a. Extensions:
        - Python
        - Docker
3. Install Docker Desktop
4. Run command in Visual Studio Code:
    python -m pip install --upgrade pip
5. Install dependencies using the following commands in the VS Code integrated terminal:
    python -m pip install flask
    python -m pip install boto3
    python -m pip install requests
    python -m pip install flask_cors
6. Create access keys for an IAM user:
    a. Sign into the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
    b. In the navigation pane, choose **Users**.
    c. Choose the name of the user whose access keys you want to create, and then choose the **Security credentials** tab.
    d. In the **Access keys** section, choose **Create access key**.
    e. To view the new access key pair, choose **Show**. You will not have access to the secret access key again after this dialog box closes. Your credentials will look something like this:
        - Access key ID: [IAM_Access_Key_ID]
        - Secret access key: [IAM_Secret_Access_Key]
7. Install AWS CLI
8. Configure AWS credentials:
    a. Run: aws configure
    b. Enter access key ID – [IAM_Access_Key_ID]
    c. Enter secret access Key – [IAM_Secret_Access_Key]
    d. Enter default region name – us-east-1
    e. Enter default output format – json

Run Frontend (Docker)

1. Open project '**app**' folder in VS Code.
2. Right Click **'Dockerfile'** in **app/backend** and select **'build image'**
3. Docker Desktop:
    a. '**Run**'
    b. Local Host: 8088
    c. Host Path: app/frontend
    d. '**Run**'
    e. Result: http://localhost:8088



Run Frontend (VS Code)

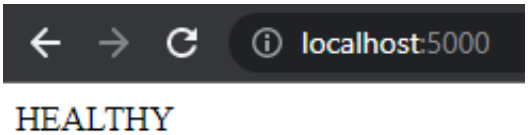
1. Open project '**app**' folder in VS Code.
2. Open terminal and run – cd frontend
3. Run - flask run --port 8088
4. Go to - http://localhost:8088/

Run Backend (Docker)

1. Open project '**app**' folder in VS Code.
2. Right Click **'Dockerfile'** in **app/backend** and select **'build image'**
3. Docker Desktop:
    a. '**Run**'
    b. Local Host: 5000
    c. Host Path: app/backend
    d. '**Run**'

Run Backend (VS Code)

1. Open project '**app**' folder in VS Code.
2. Open terminal and run – cd backend
3. Run - flash run
4. Go to - http://localhost:5000/
5. Result:



Testing AWS Credentials in AWS CLI

1. Run PowerShell as an Administrator
2. Run: aws sts get-caller-identity | tee
3. Enter: credentials.json

PRODUCTION

Prerequisites

1. Install EB CLI:
    a. Run all in elevated PowerShell:
    b. Install **virtualenv**: python -m pip install --user virtualenv
    c. Run - git clone https://github.com/aws/aws-elastic-beanstalk-cli-setup.git
    d. Run - python .\aws-elastic-beanstalk-cli-setup\scripts\ebcli_installer.py
    e. Run (Replace [Username] with user path) - & "C:\Users\[Username]\.ebcli-virtual-env\executables\path_exporter.vbs"

Create table in AWS DynamoDB

Run in AWS CLI

1. Run:

```
aws dynamodb create-table \
  --table-name UserCheckin \
  --attribute-definitions \
    AttributeName=Username,AttributeType=S \
  --key-schema AttributeName=Username,KeyType=HASH
  --provisioned-throughput ReadCapacityUnits=5,WriteCapacityUnits=5 \
  --table-class STANDARD
```

2. Run: aws dynamodb list-tables

```
PS C:\WINDOWS\system32> aws dynamodb list-tables
{
    "TableNames": [
        "UserCheckin"
    ]
}
```

Management Console:

1. Log into AWS Management Console
2. Search for **DynamoDB**
3. Select '**Create Table**'
4. Enter table name – **UserCheckin**
5. Enter partition key name – Username (**string**)
6. Select 'Customize settings'
7. Choose table class – **DynamoDB Standard**
8. Capacity mode – **Provisioned**
9. Read capacity:
    a. Auto scaling – **On**
    b. Minimum capacity units – **1**
    c. Maximum capacity units – **10**
    d. Target utilization (%) – **70**
10. Write capacity:
    a. Auto scaling – **On**
    b. Minimum capacity units – **1**
    c. Maximum capacity units – **10**
    d. Target utilization (%) – **70**

11. Encryption at rest – **Owned by Amazon DynamoDB**
12. Select **'Create Table'**
13. Done

AWS Elastic Beanstalk

Deploy Python Flask application to AWS Elastic Beanstalk to host both the frontend and backend.

1. Initialize EB CLI with the following command: eb init -p python-3.8 e2e_frontend --region us-east-1
2. Setup SSH & KeyPair:
    a. Run: eb init
    b. Setup SSH: Y
    c. Assign existing KeyPair
3. Create environment: eb create e2e-env
4. Open – eb open

Test

Test Number	Test Step	Test Data	Expected Outcome	Actual Outcome
1	Check-In a user	test, test2, test3	Check In User was checked in successfully	**Passed**
2	Return		Returns to Welcome page	**Passed**
3	Check-Out a user	test	Check Out User was checked out successfully	**Passed**
4	Return		Returns to Welcome page	**Passed**
5	Show All Users		Displays (Username/Check-In/Check-Out) table as seen in the result figure below.	**Passed**

Result

Username	Checked-In	Checked-Out
test3	true	false
test2	true	false
test	false	true

Additional Considerations

Security

- Instance security groups (Created with Elastic Beanstalk deployment)
- Load balancer security group
- VPC
- IAM
- AWS Certificate Manager
- AWS-Vault Encryption

Availability/Redundancy

- EC2 Load Balancers (Created with Elastic Beanstalk deployment)
- AWS Backup

Scalability

- Auto Scaling group (Created with Elastic Beanstalk deployment)
- Amazon CloudWatch alarms (Created with Elastic Beanstalk deployment)

References

Python Flask - https://code.visualstudio.com/docs/python/tutorial-flask

https://techobservatory.com/how-to-run-flask-in-visual-studio-code/

DynamoDB - https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/DynamoDBLocal.DownloadingAndRunning.html#DynamoDBLocal.DownloadingAndRunning.title

Boto3 - https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/dynamodb.html

https://hands-on.cloud/introduction-to-boto3-library/

Chocolatey - https://jcutrer.com/windows/install-chocolatey-choco-windows10

Elastic Beanstalk Deployment - https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/create-deploy-python-flask.html

Virtualenv - https://virtualenv.pypa.io/en/latest/installation.html

AWS Elastic Beanstalk CLI - https://github.com/aws/aws-elastic-beanstalk-cli-setup