**20CS3234AA – ADC**

**LAB EXPERIMENT GUIDANCE**

**Department of Computer Science and Engineering (Honors)**

**Experiment 1**

**Guidance PDF:** <https://aws-tc-largeobjects.s3-us-west-2.amazonaws.com/DEV-AWS-MO-GCNv2/lab-1-account.html>

**Experiment 2:**

**Guidance Video Link:** <https://www.youtube.com/watch?v=oFSU6rhFETk>

**Lambda Nodejs code to Write in DynamoDB:**

const AWS=require('aws-sdk');

const ddb=new AWS.DynamoDB.DocumentClient({region:'us-east-1'});

exports.handler=async(event,context,callback)=>{

const requestId=context.awsRequestId;

await createMessage(requestId).then(()=>{

callback(null,{

statusCode:201,

body:'',

headers:{

'Access-Control-Allow-Origin':'\*'

}

})

}).catch((err)=>{

console.error(err)

});

};

function createMessage(requestId){

const params={

TableName:'Sample',

Item:{

'id':requestId,

'message':'hello balajee'

}

}

return ddb.put(params).promise();

}

**For Reference, Nodejs Read Code:**

const AWS=require('aws-sdk');

const ddb=new AWS.DynamoDB.DocumentClient({region:'us-east-1'});

exports.handler=async(event,context,callback)=>{

await readMessage().then(data=>{

data.Items.forEach(function(item){

console.log(item.message)

});

callback(null,{

statusCode:201,

body:data.Items,

headers:{

'Access-Control-Allow-Origin':'\*',

},

})

}).catch((err)=>{

console.error(err);

})

};

function readMessage(){

const params={

TableName:'Sample',

Limit:10

}

return ddb.scan(params).promise();

}

**DynamoDB Table Creation**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Experiment 3:**

Step 1: Go to S3 bucket in Services

Step 2: Create the bucket with ACL enabled for Public Access

Step 3: Upload the object

Step 4: Make the object as “Make Public via ACL”

Step 5: Open the object link in browser tab

Step 6: Make “Static Website” enabled in created bucket with starting page like “index.html”

Step 7: upload files like “index.html”

Step 8: Go to Cloud Front and link the bucket

Step 9: Use cloud front and copy the link, attach the object name with the link copied and open it in browser tab.

**Experiment 4:**

Step 1:Go to RDS in aws

Step 2: Create MySQL based RDS by following below

Step 2.1: Choose **free tier lable**

Step 2.2: Ensure you going with **db.t2.micro**

Step 2.3: Give password (simple one)

Step 2.4: Enable Public Access

Step 2.5: Under advanced option, give the database name as “mydb”

Step 2.6: Create database

Step 2.7: After creation, change its inbound rule to All traffic IPV4

Step 2.8: Use the host, port, default user name (admin), password to connect with your workbentch or Apllication

**Demo Application – If Required:** <https://github.com/balajee-rm/aws/blob/main/demo.zip>

**Note:** If you are using my application follow these commands

In CMD (AWS cmd will be recognized, only if you are having AWS CLI2 Installed)

1. aws configure --profile aws-devops
2. use iam user credentials downloaded as a csv file

// bucket creation – replace bucket name and region name

1. aws s3 mb s3://BUCKET-NAME --region ap-south-1 --profile aws-devops

// Copy the demo zip file to the bucket created, the cmd should be in the location where your demo.zip located – replace your bucket name and region

1. aws s3 --region ap-south-1 cp demo.zip s3://BUCKET-NAME/demo --acl public-read

Go to the bucket you created just now, manually and set the bucket policy

|  |
| --- |
| { |
|  | "Version": "2012-10-17", |
|  | "Statement": [ |
|  | { |
|  | "Sid": "AllowS3Access", |
|  | "Effect": "Allow", |
|  | "Principal": { |
|  | "AWS": "arn:aws:iam::793451431447:role/S3DynamoDBFullAccessRole" |
|  | }, |
|  | "Action": "s3:\*", |
|  | "Resource": [ |
|  | "arn:aws:s3:::BUCKET-NAME", |
|  | "arn:aws:s3:::BUCKET-NAME/\*" |
|  | ] |
|  | } |
|  | ] |
|  | } |

**In Putty terminal window,**

1. sudo -apt update
2. sudo apt install unzip
3. wget <https://BUCKET-NAME.s3.ap-south-1.amazonaws.com/demo>
4. unzip demo
5. sudo apt install python3
6. sudo apt install python3-pip
7. pip install flask
8. pip install mysql-client
9. pip install mysql-connector-python
10. sudo nano task.py (for editing)

// replace this line

app.run (debug = True, host='0.0.0.0', port=8080);

//Change the connection values for RDS

1. python3 task.py

IN EC2, Allow all inbound traffic

Open with IPv4\_address:8080

**Experiment 5:**

**Teaching and Guidance Link:** <https://www.linkedin.com/learning/devops-with-aws>

**Section 1: CodeCommit**

**Free Tier Limits:**

1,000 repositories per account; up to 25,000 upon request

50 GB-month of storage

10,000 Git requests/month

**Support File Link:**

<https://github.com/balajee-rm/aws/blob/main/Session%2014%20and%2015/demo.zip>

Support files can also be downloaded from the exercises itself

**Experiment 6:**

Teaching and Guidance Link: <https://www.linkedin.com/learning/devops-with-aws>

**Section 2: CodeBuild**

**Free Tier Limits:**

100 build minutes of build.general1.small per month

**While creating codebuild, you can ensure the following**

A screenshot of a computer

Description automatically generated

**I had set the cut off time to 5 min**

**3GB Memory – is only the free tier one**

**Support File Link:**

<https://github.com/balajee-rm/aws/blob/main/Session%2014%20and%2015/demo.zip>

Support files can also be downloaded from the exercises itself

**Experiment 7:**

Teaching and Guidance Link: <https://www.linkedin.com/learning/devops-with-aws>

**Section 3: CodeDeploy**

**Free Tier Limits:**

no additional charge for code deployments to Amazon EC2, AWS Lambda or Amazon ECS through AWS CodeDeploy

**Support File Link:**

<https://github.com/balajee-rm/aws/blob/main/Session%2014%20and%2015/demo.zip>

Support files can also be downloaded from the exercises itself

**Experiment 8:**

Teaching and Guidance Link: <https://www.linkedin.com/learning/devops-with-aws>

**Section 4: CodePipeLine**

**Free Tier Limits: There should be only one active pipeline as per free tier across all regions**

Support files can also be downloaded from the exercises itself

**Support File Link:**

<https://github.com/balajee-rm/aws/blob/main/Session%2014%20and%2015/demo.zip>