1] Launch two instances of AWS EC2, one windows another ubuntu . Get connected to instances using RDP and MobaXterm client software. Explain each step of EC2 creation and launching with the help of screenshots. Open google.com from both the instances,search your own name.

1] Steps for windows

- AWS Management Console Dashboard
- Click on Services.
- Now CLICK on Compute -> EC2.
- Launch Instance
- Select Windows
- Create a key pair. A .pem file will be downloaded which will be later used to connect to the instance.
- Network Settings: Select 'Allow RDP Traffic from' Anywhere. Then Launch Instance.
- Success message will be shown after successful creation of instance
- RDP Connection We have to download a Remote Desktop File to connect to the instance.
- Go to RDP Client to download the Remote Desktop File Using the .pem file downloaded earlier, decrypt the password and copy it.
- Now, run the Remote Desktop File and put the decrypted password copied earlier
- Your Windows Instance will start running
- Open google.com to search your name.
- Terminate the instance if you don't wish to use it again.

2] Steps for ubuntu

- AWS Management Console Dashboard
- lick on Services.
- Now CLICK on Compute -> EC2.
- Launch Instance
- Select Ubuntu
- Create a key pair. A .pem file will be downloaded which will be later used to connect to the instance.
- Network Settings: Select 'Allow SSH Traffic from' Anywhere. Then Launch Instance
- Success message will be shown a er successful creation of instance
- Launch Ubuntu instance to get the remote host and username for SSH in MobaXterm.
- Go to SSH Client to get the remote host and username
- Copy the Public DNS which is your remote host and the username is the word before '@'
- Download MobaXterm to connect the instance
- Run MobaXterm. Go to Sessions → New Session.
- Then select SSH Fill the basic SSH settings and attach the .pem file downloaded earlier in advanced SSH settings 'Use private key' section →Then OK.
- Your Ubuntu Instance will be running Execute a few commands as follows – [sudo su sudo apt update].
- Create a directory of your name
- Terminate the instance if you don't wish to use it again

2] AWS CLOUD 9

- AWS Management Console Dashboard.
- Search for Cloud9 and select it.
- Click on “Create Environment”
- Name your environment and configure the settings.
- Now the AWS Cloud9 Development Environment is being Created.
- **FOR PYTHON**
    - Install Python. Run the yum update for Amazon Linux to help ensure the latest security updates and bug fixes are installed: sudo yum -y update Install Python by running the install command. For Amazon Linux: sudo yum -y install python3 Here, python is already installed so we will check the python version by running the following command: python –version
    - Add code In the AWS Cloud9 IDE, create a file with the python code and save the file with some name.
    - Run the code In AWS Cloud9 IDE, on the menu bar choose Run -> Run Configurations -> New Run Configuration.
    - On the [New] - Stopped tab, enter filename.py 15 10 for Command
    - Choose Run
    - Install and configure the AWS SDK for Python (Boto3) Install pip: sudo python3.7 get-pip.py Install the AWS SDK for Python (Boto3) – After you install pip, install the AWS SDK for Python (Boto3) by running the command: sudo python36 -m pip install boto3 Check the Boto3 version by running the following command python -m pip show boto3
    - Add AWS SDK code Add code that uses Amazon S3 to create a bucket, list your available buckets, and optionally delete the bucket you just created In the AWS Cloud9 IDE, create a file with the code content and save the file with some name.
    - Run the AWS SDK code On the menu bar choose Run -> Run Configurations -> New Run Configuration
    - For Command, enter filename.py ‘name of bucket’ us-west-2, where us-west-2 is the ID of the AWS Region where your bucket is created. By default, your bucket is deleted before the script exits
    - Choose Run
- **FOR Node.js:**
    - Install required tools Run the yum update for Amazon Linux to help ensure the latest security updates and bug fixes are installed: sudo yum -y update Run this command to install Node.js: nvm install v16.0.0 Here, node.js is already installed.
    - Add code In the AWS Cloud9 IDE, create a file with the node.js code and save the file with some name.
    - Run the code In AWS Cloud9 IDE, on the menu bar choose Run -> Run Configurations -> New Run Configuration.
    - On the [New] – Idle tab, enter filename.js 15 10 for Command
    - Choose Run
    - Install and configure the AWS SDK for JavaScript in Node.js To install the AWS SDK for JavaScript(V2) in Node.js Use npm to run the install command: npm install aws-sdk.
    - Add AWS SDK code In this step, you add some more code, this time to interact with Amazon S3 to create a bucket, list your available buckets, and then delete the bucket you just created. In the AWS Cloud9 IDE, create a file with the code content, and save the file with some name.
    - Run the AWS SDK code Enable the code to call Amazon S3 operations asynchronously by using npm to run the install command: npm install async
    - On the menu bar choose Run -> Run Configurations -> New Run Configuration
    - For Command, type filename.js ‘name of bucket’ us-east-2, where us-east-2 is the ID of the AWS Region you want to create the bucket in
    - Choose run
- Finally, close all terminals and delete the Cloud9 Environment.

3] Deployment of static website on AWS S3

- AWS Management Console Dashboard.
- Click on Service -> Storage -> S3
- Click on “Create a Bucket”.
- Give name to your Bucket.
- Uncheck the “Block all Public Access” checkbox.
- Select the Bucket
- Permission -> Edit the Bucket policy section
- Edit bucket policy

```
{
    "Version": "2008-10-17",
    "Id": "PolicyForPublicWebsiteContent",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::enter the name of your bucket/*"
        }
    ]
}
```

- Click on save changes
- Properties -> Edit “Static Website Hosting”-> Select “Enable Statics Website Hosting”.
- In the objects section attach the html code files.
- Now the html code file has been uploaded successfully.
- Now Copy the link from the Static website hosting then Paste the URL in a web browser and the web page will be displayed.
- After closing the web page, go to buckets and delete all objects from the bucket.
- Then select the empty BUCKET and delete it.
- After deleting the Bucket if we try loading the web page, we will get the 404 ERROR

4] Create a Table in DynamoDB , Add items to the table (minimum 10 items) , Query the table .

- AWS Management Console Dashboard
- Search for DynamoDB and select it.
- Click on ‘Create Table’
- Name your table→set ‘Partition key’ & ‘Sort key’→select Default setting→Click on Create Table.
- Table Created Successfully! From here select the blue link name as “Student_table”.
- Click on “Explore table item”→” then Create item”
- As 1 item is created same way we need to create 9 more items
- Add new attributes if required→Click on “Create item”.
- After entering each item select the “Create item” button and repeat the process until you are finished.
- Run a query for needed items using the partition key.
- Finally, Select the table and click on “Delete”

5] To create a Lambda function which will log “An Image has been added” once you add an object to a specific bucket in S3.Use AWS Lambda blueprint.

- AWS Management Console Dashboard.
- Click on Services→Storage→S3
- Click on “Create Bucket”
- Give your bucket a name→Click on Create bucket.
- Bucket successfully created.
- Search for ‘Lambda’ and click it
- Click on ‘Create Function’.
- Choose “Use a blueprint”→Search s3 in “Blueprints”→Select “s3-get-object python”.
- Name the Function & Role→Select your S3 bucket as trigger →Click on Create.
- Click on your function→Go to code→Add a print statement in code→Save the code and deploy it.
- Code successfully uploaded / deployed.
- Go to your bucket →Upload an image as an object in your bucket→Click on upload.
- Now the ‘Image’ is successfully uploaded .
- After the image is been uploaded you can check the message by Click the blue link in Log stream→ “Image file uploaded”.
- Now Delete your ‘Lambda Function’
- Delete and close the function .
- Empty your Bucket→Delete the object and then delete the Bucket.

6] Deploy a containerized web Application on AWS EC2 Linux. [install Docker ,pull nginx image and run it].Pull python images and run the command to list all the locally stored docker images .

- AWS Management Console Dashboard
- Search for IAM and select it
- Go to ‘Roles’ and click on ‘Create Role’.
- Select Trusted Entity ‘AWS services’ use common cases as EC2 and click to the NEXT button.
- Now in Permissions Policies → select Amazon SSM Manage. Container examples: Containers via Docker, PhotonOS, RancherOS.
- Now give name to your Role → then click on “Create Role”.
- Role Created Successfully!!
- Now go to → Services and search →EC2.
- Click on “Launch Instances” in ‘EC2 Dashboard’.
- Now give name to Your Instance.
- Select Amazon Linux.
- Click to all ‘Checkbox’
- Create a Key Pair.
- Instance Created Successfully! →Now View all Instances.
- Click on Instance ID
- Go to Connect Instance → SSH Client → copy Public DNS
- Go to “MOBAXTERM” → Session → SSH → Advance SSH Settings fill all details then ‘click OK’
- Connected to TERMINAL.
- Click to “sudo su” then →”yum install docker”
- Installation Completed
- Run command → “service docker start” then → “docker pull nginx”
- Command “docker images” then → “docker run -p 80:80 nginx”
- Go to EC2 again.
- Click on Instance ID and → copy Public IPv4 DNS.
- Go to Google and Paste the Link
- Welcome to nginx!
- Run command → “docker ps -a”
- Now close the tab
- Terminate Instances.
- Now Delete the Role in IAM .

7] DOCKER HUB AND ITS COMMANDS ON AWS

- login to AWS Management Console Dashboard.
- Search for EC2 and select it.
- Go to EC2 Dashboard and “Launch Instances”.
- Launch an Instance and then ‘Give name to your instances’.
- Select UBUNTU and 20.04 LTS Server.
- Create a Key Pair.
- Allow all the HTTPs Traffic from the Internet and click to ‘Launch Instance’.
- Instances Created Successfully. Now view the instances.
- Connect the Instances, Go to SSH client and copy public DNS.
- Go to ‘MobaXterm’→Session →SSH →Adv SSH Setting then allow it.
- Install docker on AWS EC2 –Ubuntu by using curl
- Run the command ‘sudo su’ to gain root user access. Then enter commands:
    - curl -fsSLhttps://get.docker.com -o get-docker.sh
    - sh get-docker.sh
- Enter command ‘docker -- version’ to see the current docker version. Then run command ‘docker images’ to see installed images. Also run the command ‘docker run hello-world’ which will run helloworld from docker.hub.com and run it.
- Now run ‘docker images’ again, the repository will have an image named as ‘hello-world’.
- Run command ‘docker run -it -p 80:80 –name[name of your webpage] nginx bash’. Now to launch the nginx web server, copy the Ipv4 address from the EC2 instance details and paste it to the address bar of Web Browser
- Can’t able to Reach the site.
- Within the container use command ‘service nginx start’ to deploy the web server.
- After deploying the web server, the web page will be visible without any errors.
- Take a duplicate tab run command ‘sudo su’ →‘docker ps’ →‘docker ps -a →‘#docker execute -it 279e bash’.
- To make changes to a file within the container use command ‘docker exec - it(container id bash)’
- Run ‘docker run -it -p 80:80 -name[name of your webpage] nginx bash’ → ‘service nginx start’. Update the container using “apt update”.
- Now within the coninter navigate to the html directory using command: ‘cd usr/share/nginx/html’. Then update the container using [apt update].
- Now use the command ‘apt install nano’ to install nano text Editor.
- Now move the original nginx index as a backup so you can create your own html. Index file using command ‘mv index.html index.html.backup’. Then, open the nano text editor using the command ‘nano index.html’.
- Now run command ‘docker ps’ to get the ‘container id’ and run ‘docker stop (container id)’ to stop the running container.
- **Pull 3 or 4 images, one of the python, and run “Hello World ” inside the container.**
- Run ‘docker --version’ to get the version and also run ‘docker images’ to check the images uploaded in the repository.

- Now pull mysql from ‘docker.hub.com’ to install the mysql image. Then run command ‘docker images’ to check the images uploaded in the repository.
- Same like mysql, now pull python from ‘docker.hub.com’ to install python image. Then run command ‘docker images’ to check the images uploaded in the repository
- Now, to get access to the python container use command→ ‘docker run -it (images id) bash’ and type ‘python’ to enter the shell and write any code.
- Enter command ‘docker --version’ to see the current docker version.
- Then enter command ‘docker python --version’ to see the current python version.
- Same command ‘docker mysql --version’ to see the current mysql version.
- And then enter ‘docker nginx --version’ to see the current nginx version.
- For checking the status of the container use command ‘docker ps -a’.
- Run command ‘docker inspect nginx’ to inspect the nginx.
- Now run ‘docker inspect mysql’ to inspect mysql.
- Now run command ‘docker history python’ to get the docker python history
- Now stop the container by using command ‘docker stop [container id]’ → and simultaneously kill the docker by ‘docker kill’
- Now exit the Session / Terminal.
- **Demonstrate any 15 docker commands and explain its uses**.
- After running all the codes, now Terminate the Instance in ES2. Click on Terminate to ‘Terminate the Instance’.

8] Install docker on AWS EC2 –Ubuntu by using curl

- login to AWS Management Console Dashboard.
- Search for EC2 and select it.
- Click on “Launch Instances”.
- Give any name to your “Instance”.
- Select UBUNTU and go with 20.04 LTS(HVM), SSD Volume Type.
- Create a KeyPair.
- Allow all the Traffic from the Internet and click on Launch Instances. Click on View Instances.
- Connect to the Instance.
- Go to the SSH client and copy Public DNS.
- Now paste the Public DNS in MobaXterm for that Go to MobaXterm → Session → SSH →Adv SSH Setting.
- Your UBUNTU terminal is created, for Root access type “sudo su”.
- Now use the command “curl -fsSL https://get.docker.com -o get-docker.sh” & “sh get-docker.sh” to pull Docker.
- **Run a Flask Application inside a Docker Container and explain the steps.**
- Run command “docker --version” then “docker images”.
- Now create a directory [your own name], go inside it and create another directory name as “APP” and go inside it too.
    - Mkdir your name
    - Cd your name
    - Mkdir app
    - Cd app

- Now open the nano editor name by [nano app.py] and add the code of flask given in the classroom.
    - From flask import flask
    - app= flask(__name__)
    - @app.route(‘/’)
    - Def hello():
        Return “Demo flask & docker application is up and running!”
      If __name__ = ‘__main__’
        app.run(host = “0.0.0.0”, port =80)
- Now open a nano editor named by [requirements.txt] and write the requirements of the file to be installed on another PC.
    - Flask
- Now go back to your Parent Directory and write [nano Dockerfile] and copy the code give in the classroom.
    - Cd..
    - Nano Dockerfile
    - Docker built -t your name:latest
- Write code in Dockerfile.

    - Sudo su
    - Mkdir your_name
    - Cd your_name
    - mkdir app
    - Cd app
- Now the command “docker build -t [file name]:latest .”
- Now for checking the images run the command as “docker images”.
- Run command “docker run -d -p 80:80 [directory name]”.
- To run the above command, copy the Public IPv4 DNS.
- Paste it in the Address Bar
- After performing, terminate the instance.
- Click on Terminate, Instance will be Terminated.
- Simultaneously, delete the Key Pair too.

9] Deploy a web Application [any language] using AWs Elastic beanstalk.

- Login to AWS Management Console Dashboard.
- Search for Elastic Beanstalk and select it.
- Click on Create Application.
- Give name to your ‘Application’ → ‘Create Application’.
- Now the application is created within 5 to 10 mins.
- The Application has been created in “Java”.
- Click on the URL given.
- Congratulations, your Java Application has been created.
- Now search for EC2 and select it.
- Click on Instance.
- This is the influence of Elastic Beanstalk on EC2 and now the instance is running
- We can see all the details of the instance.
- Now terminate the Instances.
- Now go to → ‘ELASTIC BEANSTALK’ → Terminated the Environment
- Now delete the Application.

10] To understand AWS Lambda, create your first Lambda functions using Python / Java / Nodejs.

- AWS Management Console Dashboard.
- Search for “IAM” and select it.
- “IAM dashboard home page”.
- Go to role and click on “Create role”.
- Select lambda use case→ Click on next.
- Search Amazon DynamoDB in permission policies → Select policy which provides “full access to DynamoDB” → Click on next.
- Give name to your role→ Click on ‘Create role’.
- Role created successfully!
- Go to services→ click on “Lambda”
- Click on Create function.
- Choose “Author from Scratch” → Give name to your function→ Choose “Python” language in runtime.
- Permissions → Change execution role → Choose “Use an existing role” → Click on Create function.
- Function created successfully!
- Go to Services → Select S3.
- Click on “Create bucket”.
- Give name to your bucket.
- Search for Lambda and select it.
- Click on ‘Lambda Function Name’.
- Upload ‘Code’ to the Lambda Function and Deploy it.
- Then click on “Add Trigger” and add a bucket from S3 into the trigger, so that it can generate a trigger whenever something is added to the bucket.
- Select ‘S3’ add your ‘bucket’ click to the checkbox of “I Acknowledge” & then click to “Add”.
- Trigger added successfully! Also Save the code→Deploy it.
- Go to services → Select DynamoDB.
- Create the table.
- Created Table Successfully!!
- Go to S3 → Bucket and upload an object by clicking on add files.
- Successfully Uploaded
- Go to DynamoDB to check whether the items are updated in the table or not.
- DynamoDB → Tables →update settings →Explore table items.
- Again, go to S3 to upload 2 to 3 items in the table.
- Upload → add files → click on upload button.
- Again, go to DynamoDB to check whether the rest of the items are updated in the table or not.
- Repeat the steps DynamoDB → Tables →update settings →Explore table items, all items added successfully.
- Now delete ‘Role’ which was created in ‘IAM Dashboard’

- Role Deleted Successfully!!