# Assignment No 6

**Title-** Setting up AWS Environment: Create a new AWS account, Secure the root user, Create an IAM user to use in the account Set up the AWS CLI, Set up a Cloud 9 environment.

## **Name:** Thorve Avishkar S

## **Roll No.:** 63

**Steps:**

1. **Create a New AWS Account** 
   1. Go to [AWS Sign Up](https://aws.amazon.com/)
   2. Click **Create an AWS Account**.
   3. Enter your **email address**, **AWS account name**, and set a **password**.
   4. Select your **account type** (Personal or Business).
   5. Enter your **billing details** (AWS requires a valid credit/debit card).
   6. Complete **identity verification** (enter phone number, receive OTP).
   7. Choose a **Support Plan** (Free tier is recommended for learning).
   8. Log in to your AWS account using the **root user email and password**.
2. **Secure the Root User** 
   1. **Enable MFA (Multi-Factor Authentication)** o Sign in as the root user. o Go to **IAM** → **Users** → Select **Root User**. o Click **Security Credentials** → Click **Activate MFA**. o Choose **Virtual MFA device** (Google Authenticator, Authy, etc.).

o Scan the QR code with the MFA app and enter the generated codes.

* 1. **Create Billing Alerts** o Open **AWS Billing Dashboard**. o Click **Budgets** → **Create a Budget**. o Set up an alert for unexpected charges.
  2. **Do NOT use the root user for daily activities**.

1. **Create an IAM User** 
   1. Go to **IAM (Identity & Access Management)**.
   2. Click **Users** → **Add users**.
   3. Enter a **username** (e.g., admin-user).
   4. Select **AWS Management Console access** and **Generate an auto password**.
   5. Click **Next: Permissions** → Select **Attach policies directly**.
   6. Assign **AdministratorAccess** for full control or **custom permissions** for limited access.
   7. Click **Next: Tags** → (Optional) Add tags.
   8. Click **Create User** and download the credentials.

1. **Set Up AWS CLI**

Install AWS CLI

* + **Windows**: Download and install from [AWS CLI.](https://aws.amazon.com/cli/)
  + **Linux/macOS**: Run:

sh

CopyEdit

curl "https://awscli.amazonaws.com/AWSCLIV2.pkg" -o "AWSCLIV2.pkg" sudo installer -pkg AWSCLIV2.pkg -target / OR

sh

CopyEdit sudo apt install awscli -y # Ubuntu/Debian OR

sh

CopyEdit

brew install awscli # macOS with Homebrew Configure AWS CLI

1. Run:

sh

CopyEdit aws configure

1. Enter:

pgsql

CopyEdit

AWS Access Key ID [None]: <Your IAM user Access Key>

AWS Secret Access Key [None]: <Your IAM user Secret Key>

Default region name [None]: us-east-1 (or any preferred region) Default output format [None]: json (or table/text)

1. Test the setup:

sh

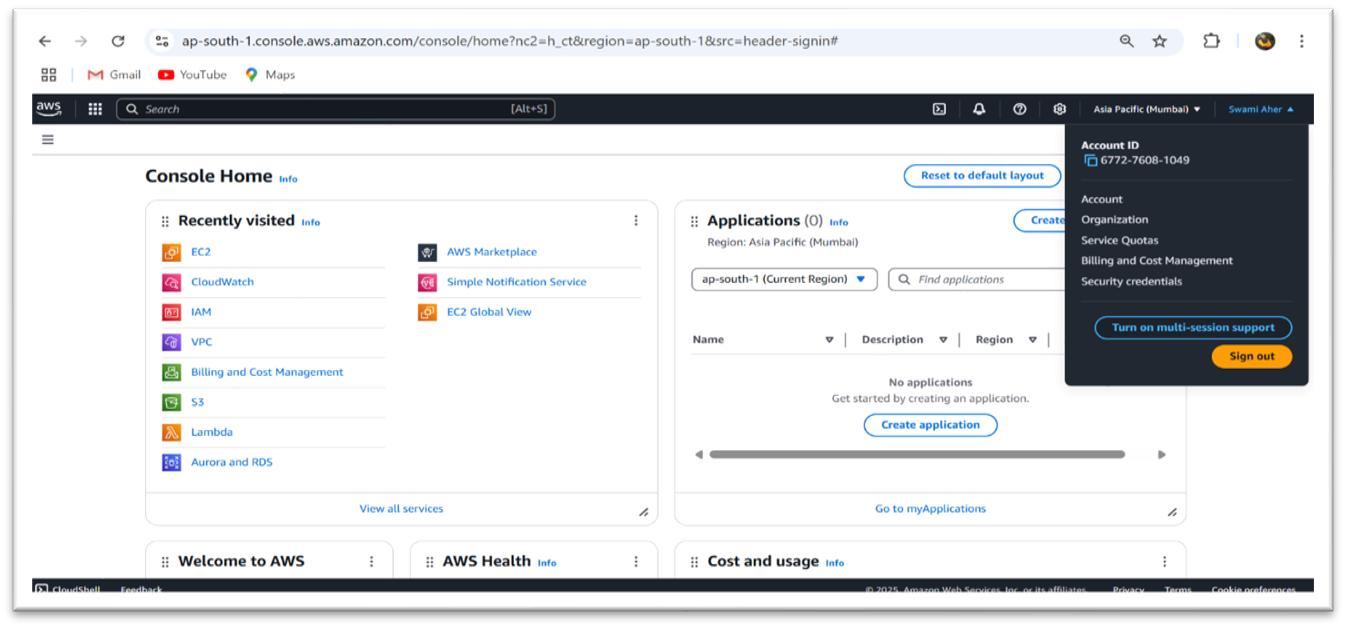
CopyEdit

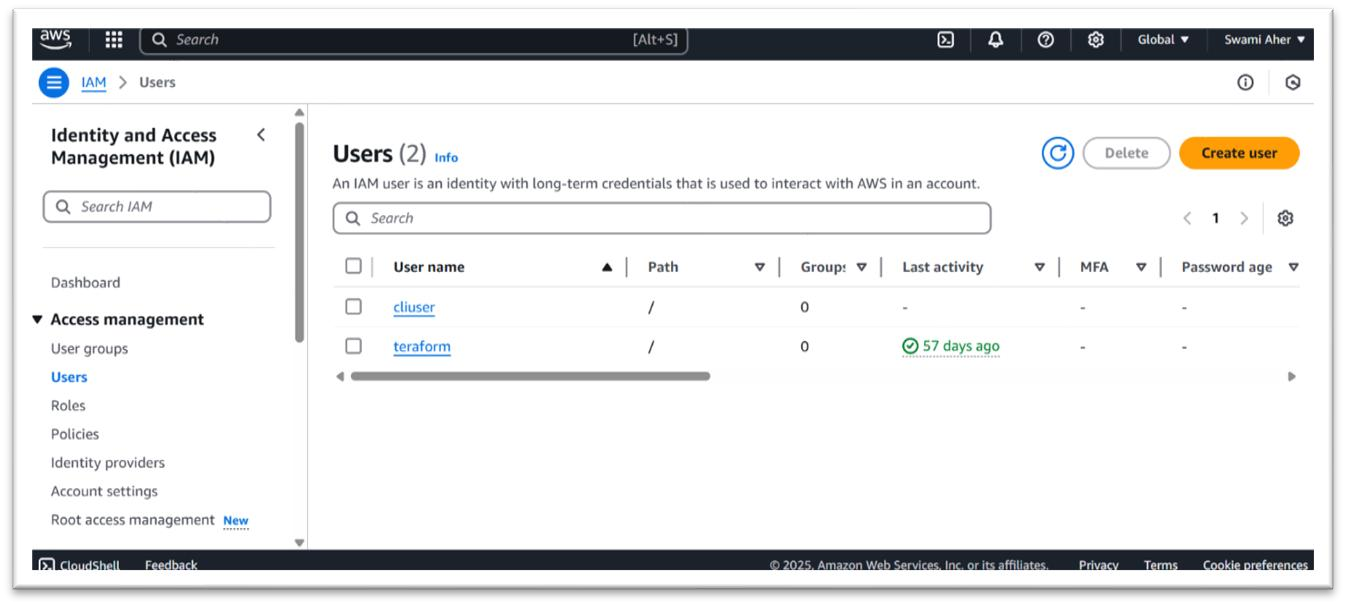
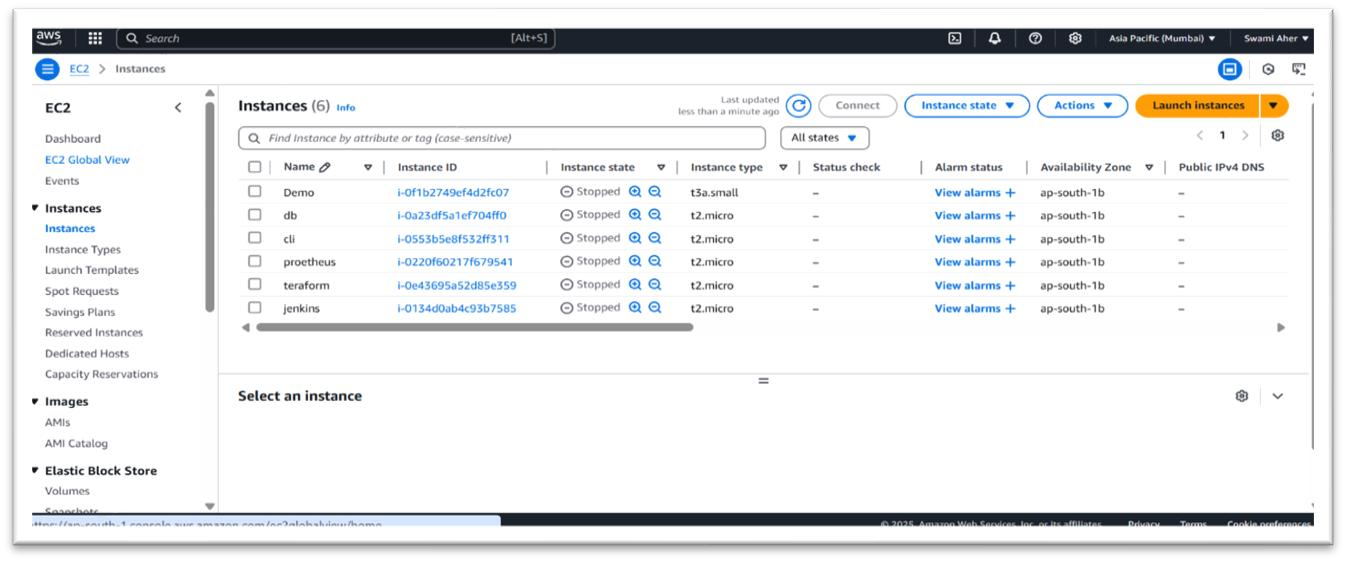
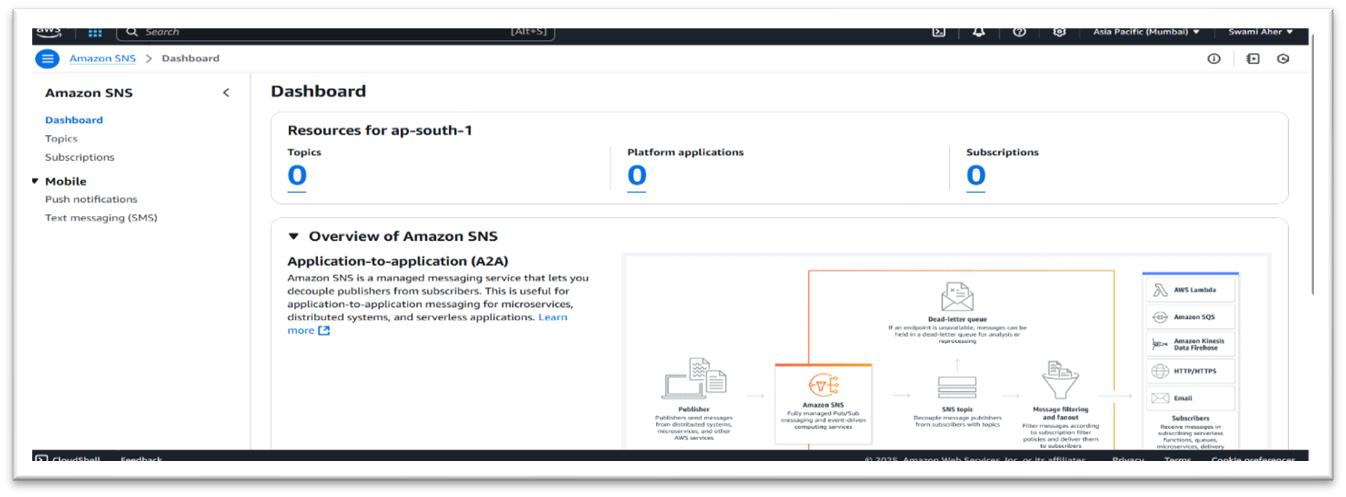
aws s3 ls

**5. Set Up AWS Cloud9 (Cloud IDE)**

1. Go to **AWS Console** → **Cloud9**.
2. Click **Create environment**.
3. Enter a name (e.g., MyDevEnvironment).
4. Choose **"Create a new EC2 instance for environment"**.
5. Select **Instance type** (default t2.micro is free-tier).
6. Choose **Networking** (default settings are fine).
7. Click **Next** → **Create Environment**.
8. Wait for the environment to be provisioned.

**Output:**





# Assignment No 7

**Title-** Setup, Create and visualize data in an Amazon Relational Database

(Amazon RDS) MS SQL Express server using Amazon Quick Sight

## **Name:** Thorve Avishkar S

**Roll No.:** 63

**Steps:**

**1. Set Up an Amazon RDS SQL Server (MS SQL Express)**

Step 1: Create an RDS Instance

1. **Login to AWS Console** → Open [Amazon RDS.](https://console.aws.amazon.com/rds)
2. Click **Create Database**.
3. **Select Engine**:
   * Choose **Microsoft SQL Server**.
   * Select **SQL Server Express** (free-tier eligible).
4. **Configure Settings**:
   * **DB instance identifier**: my-sql-express o **Master username**: admin o **Master password**: YourStrongPassword!
5. **Instance Settings**:
   * DB instance class: db.t3.micro (Free-tier eligible).
   * Storage: 20 GiB (Auto-scaling optional). o **VPC**: Select the default VPC (or create a new one).
   * **Public Access**: **Enable** (to connect externally).
6. **Additional Configurations**:
   * Parameter group: Leave default. o Backup retention: Set as needed.
   * Enable **IAM DB authentication** (optional).
7. Click **Create Database** and wait for it to be available.

Step 2: Allow Access to RDS

1. **Modify Security Group**:

* + Open **EC2 Dashboard** → Security Groups. o Find the security group attached to your RDS instance.
  + Click **Inbound Rules** → **Edit Inbound Rules**.
  + Add Rule:
    - Type: **MS SQL (TCP/1463)**
    - Source: **Your IP** (My IP option) or **0.0.0.0/0** (public, not recommended).
  + Click **Save Rules**.

Step 3: Connect to SQL Server Using SSMS or SQLCMD

1. Open **Microsoft SQL Server Management Studio (SSMS)**.
2. Use **RDS Endpoint** as the server name.
   * Format: my-sql-express.xxxxxxx.us-east-1.rds.amazonaws.com
   * Login: **admin** o Password: **YourStrongPassword!**
3. Click **Connect**.
4. Run the following SQL commands to create a sample database and table:

sql

CopyEdit

CREATE DATABASE SalesDB;

USE SalesDB;

CREATE TABLE Sales (

ID INT IDENTITY(1,1) PRIMARY KEY,

ProductName VARCHAR(255),

SalesAmount DECIMAL(10,2),

SaleDate DATE

);

INSERT INTO Sales (ProductName, SalesAmount, SaleDate)

VALUES

('Laptop', 1200.50, '2025-03-18'),

('Mouse', 25.00, '2025-03-17'), ('Keyboard', 45.99, '2025-03-16');

5. Verify data:

sql

CopyEdit

SELECT \* FROM Sales;

1. **Connect Amazon QuickSight to RDS** Step 1: Enable QuickSight Access to RDS
   1. Open [Amazon QuickSight.](https://quicksight.aws.amazon.com/)
   2. Go to **Manage QuickSight** (top right).
   3. Select **Security & Permissions**.
   4. Click **Manage VPC Connections** → **Add a VPC Connection**.
   5. Choose the **VPC where your RDS instance is deployed**.
   6. Click **Save**.

Step 2: Connect QuickSight to SQL Server

* 1. Open QuickSight → Click **Datasets**.
  2. Click **New Dataset** → Select **Microsoft SQL Server**.
  3. **Enter Connection Details**:
     + **Data source name**: SQLExpressRDS
     + **Host**: my-sql-express.xxxxxxx.us-east-1.rds.amazonaws.com
     + **Port**: 1463 o **Database name**: SalesDB o **Username**: admin o **Password**: YourStrongPassword!
  4. Click **Validate Connection** → If successful, click **Create Data Source**.

1. **Visualize Data in Amazon QuickSight**

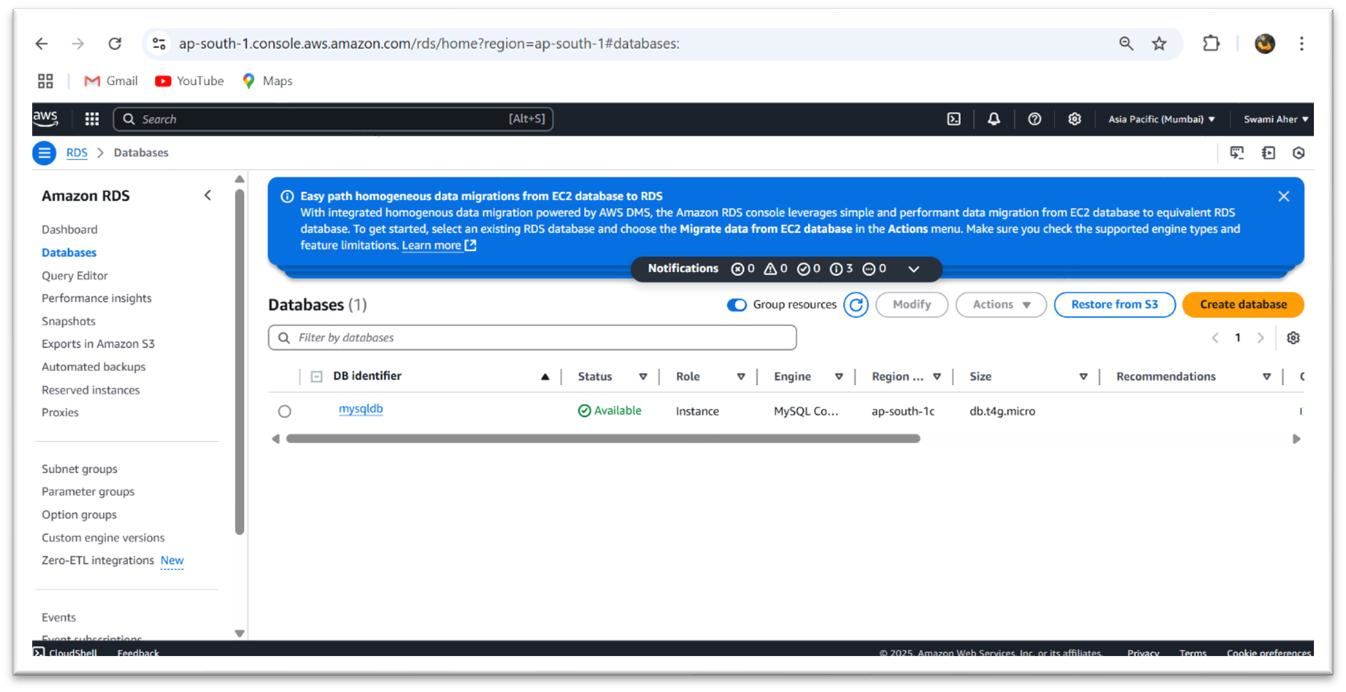
Step 1: Create a QuickSight Analysis

* 1. Choose **Sales** table → Click **Create Dataset**.
  2. Choose **SPICE** (for faster processing) or **Direct Query**.
  3. Click **Visualize**.

Step 2: Create a Bar Chart

* 1. Click **Add Visual**.
  2. Select **Bar Chart**.
  3. Drag **ProductName** to the **X-axis**.
  4. Drag **SalesAmount** to the **Y-axis**.
  5. Set **Filters** if needed.

**Output:**



# Assignment No 8

**Title-** Setup, Create and connect your Word Press site to an object storage bucket using Light sail service.

## **Name:** Thorve Avishkar S

**Roll No.:** 63

**Steps:**

1. **Set Up a WordPress Instance in AWS Lightsail**

Step 1: Create a Lightsail Instance

* 1. **Sign in to AWS** → Open [AWS Lightsail.](https://lightsail.aws.amazon.com/)
  2. Click **Create instance**.
  3. Choose **Instance location** (region closest to your audience).
  4. Select **Linux/Unix** as the platform.
  5. Under **Blueprint**, select **WordPress**.
  6. Choose an **Instance Plan**:
     + Free-tier: **$3.50/month** (512MB RAM, 1vCPU, 20GB SSD).
     + Higher tiers for better performance.
  7. Enter an **instance name** (e.g., my-wordpress-site).
  8. Click **Create Instance** and wait for provisioning.

Step 2: Get the WordPress Admin Password

* 1. Once the instance is running, click on it.
  2. Click **Connect using SSH**.
  3. Run the following command to get the WordPress admin password:

sh

CopyEdit

cat bitnami\_application\_password

* 1. Copy the password.

Step 3: Access the WordPress Site

* 1. Open your browser and go to:

arduino CopyEdit

http://<Lightsail\_Public\_IP>/wp-admin

* 1. Log in with:
     + **Username**: user
     + **Password**: (paste the copied password)

3. WordPress Dashboard should now be accessible.

1. **Set Up an Object Storage Bucket in Lightsail**

Step 1: Create a Lightsail Object Storage Bucket

* 1. Open **AWS Lightsail** → Click **Storage**.
  2. Click **Create a bucket**.
  3. Select a **Region** (same as your instance).
  4. Choose a **Bucket Name** (e.g., my-wp-media).
  5. Choose **Public or Private** (Public for direct access, Private for secure access).
  6. Select a storage plan (e.g., **5GB Free-Tier**).
  7. Click **Create bucket**.

Step 2: Create an Access Key for WordPress

* 1. Open the **Bucket settings**.
  2. Click **Access Keys** → **Create New Access Key**.
  3. Copy:
     + **Access Key ID** o **Secret Access Key**

1. **Connect WordPress to the Lightsail Object Storage Bucket**

Step 1: Install & Configure WP Offload Media Plugin

* 1. In WordPress, go to **Plugins** → **Add New**.
  2. Search for **WP Offload Media Lite**.
  3. Click **Install Now**, then **Activate**.

Step 2: Configure the Plugin

* 1. In WordPress, go to **Settings** → **Offload Media**.
  2. Click **Set Up Storage Provider**.
  3. Select **Amazon S3 Compatible Storage**.
  4. Enter:
     + **Access Key ID**: (from Lightsail) o **Secret Access Key**: (from Lightsail) o **Bucket Name**: my-wp-media
     + **Endpoint**: https://s3.<your-region>.amazonaws.com

5. Click **Save Changes**.

Step 3: Enable Media Uploads to the Bucket

1. In **Offload Media Settings**, enable:

* + - **"Automatically offload new media to bucket"**. o **"Remove from local server after offload"** (optional).

2. Upload an image in **Media Library** → It should now be stored in Lightsail Object Storage.

1. **(Optional) Point a Custom Domain to WordPress**

Step 1: Create a Lightsail Static IP

* 1. In **Lightsail**, go to your instance.
  2. Click **Networking** → **Attach Static IP**.
  3. Choose your instance and click **Create**.

Step 2: Update DNS Settings

* 1. Register a domain in **Route 53** (or any domain provider like Namecheap, GoDaddy).
  2. Add an **A record** pointing to your **Static IP**.
  3. Wait for DNS propagation.

**Output:**

