Migrating a Web Application database to Amazon RDS

JULEANNY NAVAS
AWS Cloud Computing

Introduction

In this project, I created an **Amazon RDS Maria DB** instance by using **AWS CLI.** I then migrated data from a MariaDB database hosted on an **EC2** instance running Linux, Apache, **MySQL**, and PHP (**LAMP**) to the Amazon RDS MariaDB instance. Finally, I monitored the Amazon RDS instance using **Amazon CloudWatch** metrics.

The goal was to enhance scalability, reliability, and performance while minimizing downtime. This hands-on AWS project follows a structured approach to ensure an efficient migration.

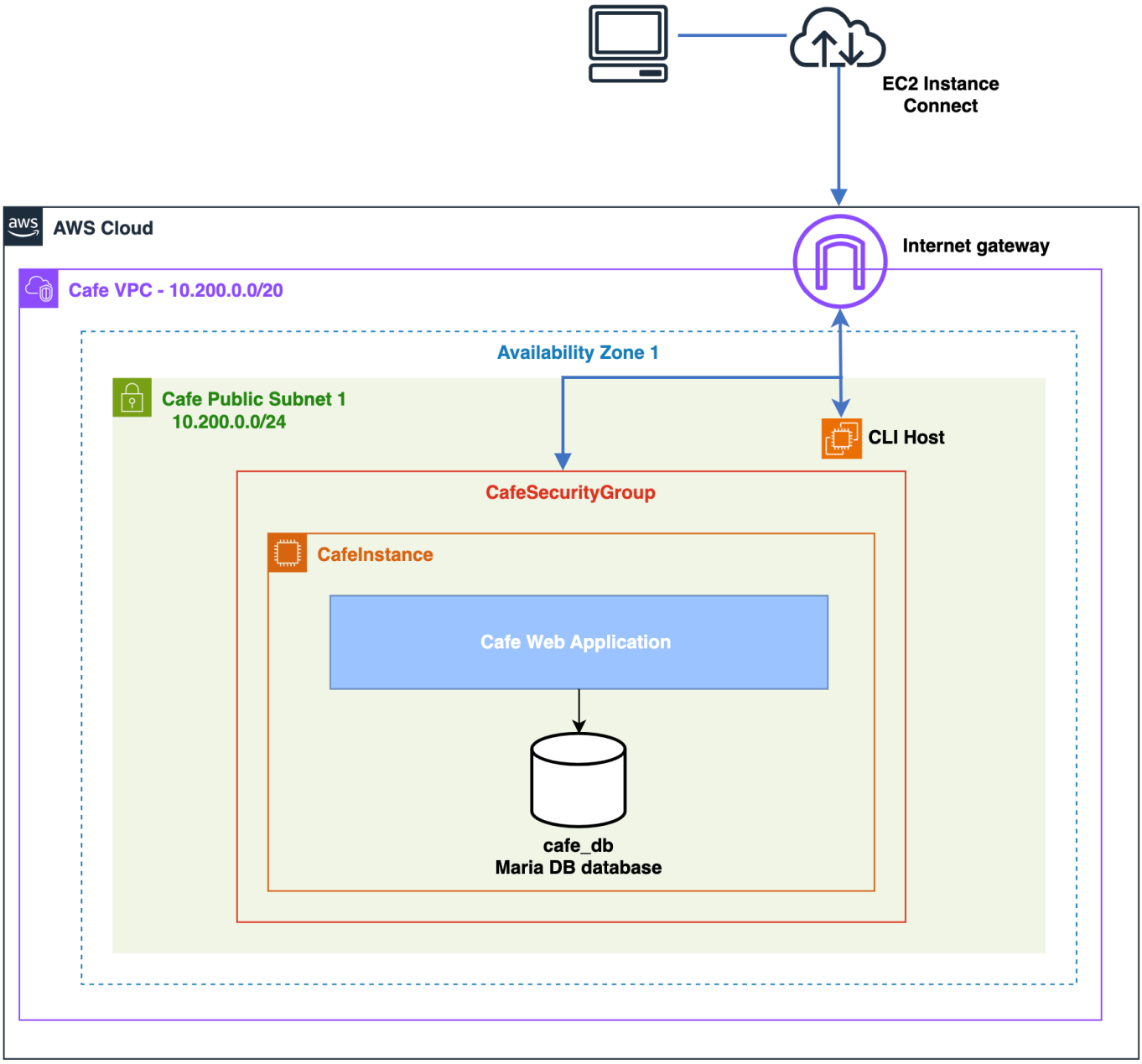
Technologies used

- **AWS RDS** (Relational Database Service)
- **AWS EC2** (Elastic Compute Cloud <Access through **Instance Connect**>)
- **AWS CloudWatch** (Monitoring)
- **AWS System Manager** (Parameter Store)
- **MySQL** (Database)
- **app.diagrams.net** (Architecture design)

Architecture Overview

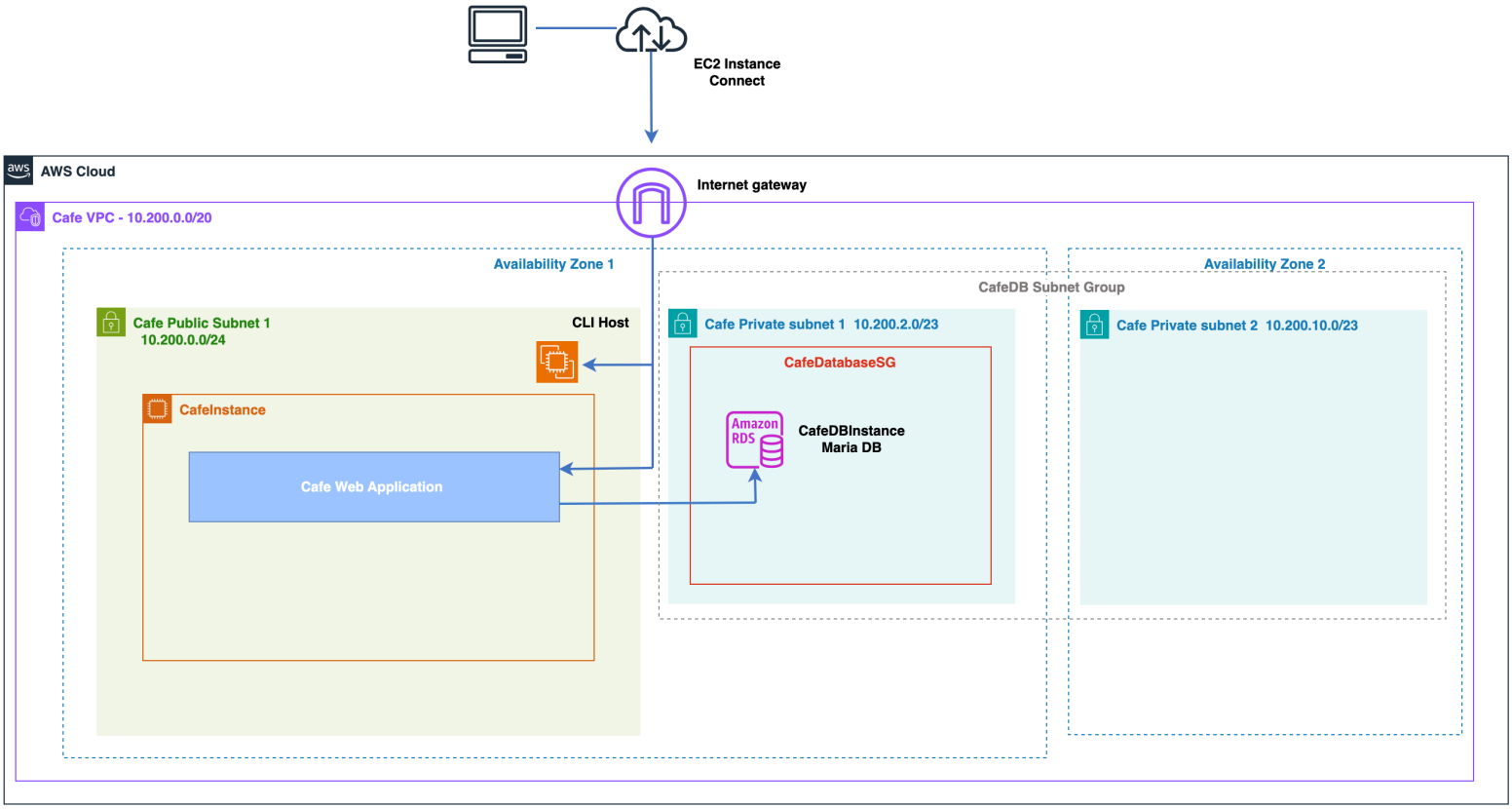
Before Migration

- The Application database was hosted on an **EC2** LAMP instance (Linux, Apache, MySQL, PHP).
- A **CLI Host** instance resides in the same subnet to facilitate the administration of the instance by using AWS CLI.
- Performance issues arouse due to manual scaling and maintenance.
- Increased operational overhead was a challenge.


EC2 Instance
Connect
AWS Cloud
Internet gateway
Cafe VPC - 10.200.0.0/20
Availability Zone 1
Cafe Public Subnet 1
10.200.0.0/24
CLI Host
CafeSecurityGroup
CafeInstance
Cafe Web Application
cafe_db
Maria DB database

After Migration

- The database is now hosted on **Amazon RDS,** residing outside the EC2 instance.
- Availability has improved, with automated scaling.
- Integrated monitoring is enable via **Amazon CloudWatch**.



Step-by-Step Implementation

1 . Generating Sample Data

I browsed the café website and placed a few orders, which were stored in the existing database:

1. In the **AWS Console,** navigate to EC2 > Instances.
2. Select **CafeInstance.**
3. Copy the **CafeInstance public URL.**
4. Paste the URL it into a new browser window and place some orders.
5. After submit your order, go to the **Order History** page and save the information for future comparison with the data in the migrated database.

Café

Home Menu Order History

Order History

Order Number: 1 **Date: 2025-03-12** **Time: 13:33:10** **Total Amount: $21.50**

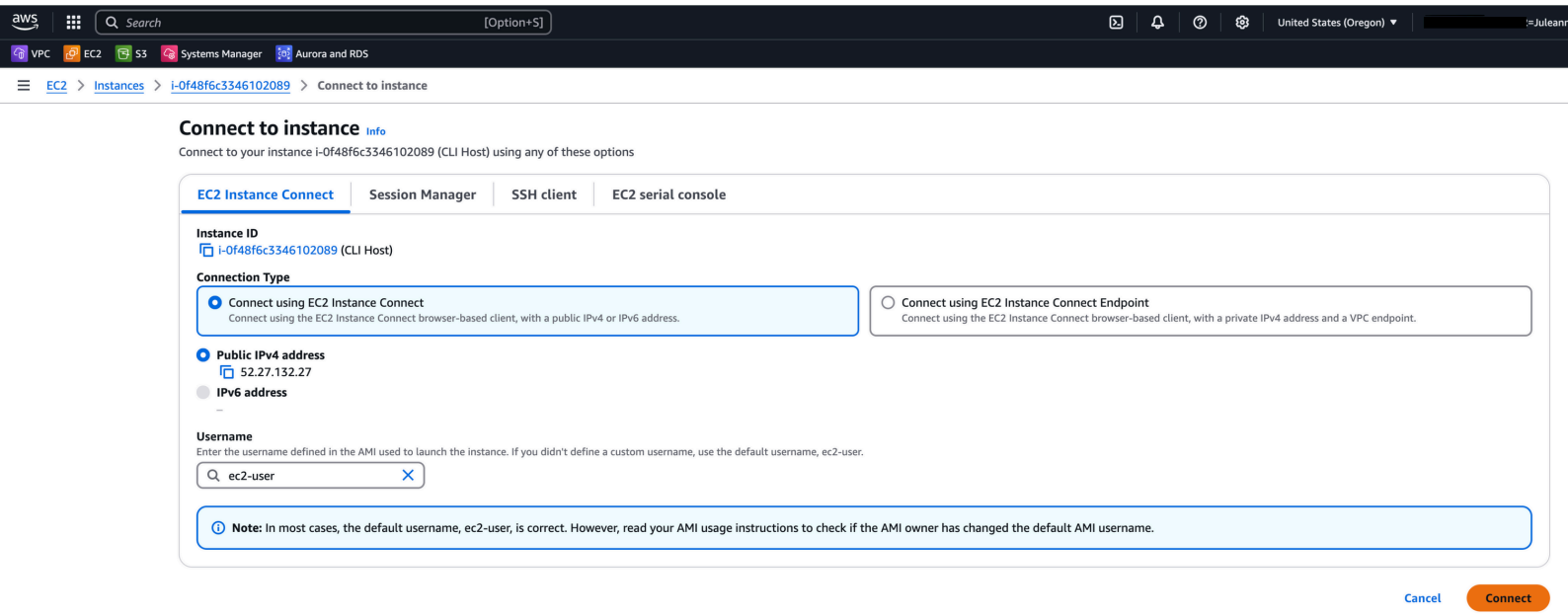
Item	Price	Quantity	Amount
Donut	$1.00	5	$5.00
Strawberry Tart	$3.50	2	$7.00
Hot Chocolate	$3.00	2	$6.00
Latte	$3.50	1	$3.50

2 . Creating an Amazon RDS Instance using AWS CLI

- **Connecting to the CLI Host instance**:

I used **EC2 Instance Connect** to access the CLI Host EC2 instance, enabling me to run AWS CLI commands.

  - Access the **EC2 Management Console** and navigate to **Instances.**
  - Select the CLI Host instance.
  - Click **Connect**, then choose the EC2 Instance Connect tab.
  - Click connect to establish the session.



- **Configuring the AWS CLI**:

I configured AWS CLI to be able of interact with AWS services, ensuring seamless execution of commands for managing resources.

```
[ec2-user@ip-          ~]$ aws configure
AWS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [None]: us-west-2
Default output format [None]: json
```

- **Creating CafeDatabase Security Group**:

This CafeDatabase Security Group was configured to allow only **MySQL** (TCP, port 3306) from instances associated with the CafeSecurityGroup. This ensures that only the CafeInstance can

access the database securely.

```
[ec2-user@ip-                ~]$ aws ec2 create-security-group \
> --group-name CafeDatabaseSG \
> --description "Security group for Cafe database" \
> --vpc-id vpc-0bdd6f3c88e6383d0
{
    "GroupId": "sg-03d1b5d5a80e155e0"
}
```

```
[ec2-user@ip-                ~]$ aws ec2 authorize-security-group-ingress \
> --group-id sg-03d1b5d5a80e155e0 \
> --protocol tcp --port 3306 \
> --source-group sg-0727e5fad3e77c530
[ec2-user@ip-1                ]$
```

Confirmed the inbound rule was applied appropriately, showing that the CafeDatabaseSG now has an inbound rule that allows connections from TCP por 3306 in the source of the connection is an instance that has the CafeSecurityGroup ID.

```
[ec2-user@ip-                ~]$ aws ec2 describe-security-groups \
> --query "SecurityGroups[*].[GroupName,GroupId,IpPermissions]" \
> --filters "Name=group-name,Values='CafeDatabaseSG'"
[
    [
        "CafeDatabaseSG",
        "sg-03d1b5d5a80e155e0",
        [
            {
                "PrefixListIds": [],
                "FromPort": 3306,
                "IpRanges": [],
                "ToPort": 3306,
                "IpProtocol": "tcp",
                "UserIdGroupPairs": [
                    {
                        "UserId": "525969271926",
                        "GroupId": "sg-0727e5fad3e77c530"
                    }
                ],
                "Ipv6Ranges": []
            }
        ]
    ]
]
```

- Creating two private subnets and a database subnet group:

Create CafeDB Private Subnet 1 and CafeDB Private Subnet 2:

This Private Subnet 1 hosts the **RDS DB instance** and is defined in the same Availability Zone as the CafeInstance. The Private Subnet 2 is required to form the Database Subnet Group, it is an empty private subnet defined in a different AZ than the CafeInstance

I assigned the CIDR (Classless Inter-Domain Routing) block that:

- Falls within the VPC's address range
- Does not overlap with any other subnet in the VPC.

Subnet Configuration:

- Cafe **VPC** IPv4 CIDR block: 10.200.0.0/20 (IP range 10.200.0.0 - 10.200.15.255)
- Cafe **Public Subnet 1**: 10.200.0.0/24 (IP range: 10.200.0.0 - 10.200.0.255)
- CafeDB **Private Subnet 1**: 10.200.2.0/23 (IP range: 10.200.2.0 - 10.200.3.255)
- CafeDB **Private Subnet 2**: 10.200.10.0/23 (IP range: 10.200.10.0 - 10.200.11.255)

```
[ec2-user@i                ~]$ aws ec2 create-subnet \
> --vpc-id vpc-0bdd6f3c88e6383d0 \
> --cidr-block 10.200.2.0/23 \
> --availability-zone us-west-2a
{
    "Subnet": {
        "MapPublicIpOnLaunch": false,
        "AvailabilityZoneId": "usw2-az1",
        "AvailableIpAddressCount": 507,
        "DefaultForAz": false,
        "SubnetArn": "arn:aws:ec2:us-west-2:525969271926:subnet/subnet-0707b61b9b3f29611",
        "Ipv6CidrBlockAssociationSet": [],
        "VpcId": "vpc-0bdd6f3c88e6383d0",
        "State": "available",
        "AvailabilityZone": "us-west-2a",
        "SubnetId": "subnet-0707b61b9b3f29611",
        "OwnerId": "525969271926",
        "CidrBlock": "10.200.2.0/23",
        "AssignIpv6AddressOnCreation": false
    }
}
```

```
[ec2-user@i                ~]$ aws ec2 create-subnet \
> --vpc-id vpc-0bdd6f3c88e6383d0 \
> --cidr-block 10.200.10.0/23 \
> --availability-zone us-west-2b
{
    "Subnet": {
        "MapPublicIpOnLaunch": false,
        "AvailabilityZoneId": "usw2-az2",
        "AvailableIpAddressCount": 507,
        "DefaultForAz": false,
        "SubnetArn": "arn:aws:ec2:us-west-2:525969271926:subnet/subnet-03e949f623dc95cbc",
        "Ipv6CidrBlockAssociationSet": [],
        "VpcId": "vpc-0bdd6f3c88e6383d0",
        "State": "available",
        "AvailabilityZone": "us-west-2b",
        "SubnetId": "subnet-03e949f623dc95cbc",
        "OwnerId": "525969271926",
        "CidrBlock": "10.200.10.0/23",
        "AssignIpv6AddressOnCreation": false
    }
}
```

Creating CafeDB Subnet Group:

The CafeDB Subnet group consists of the two private subnets in different AZ created in the previous step, it ensures:

- High Availability: RDS can deploy instances in multiple AZs for failover support.
- Private Access: The database instance remains isolated from the public internet.
- Scalability: AWS RDS can automatically choose an optimal subnet for deployment.

```
[ec2-user@ip              ~]$ aws rds create-db-subnet-group \
> --db-subnet-group-name "CafeDB Subnet Group" \
> --db-subnet-group-description "DB subnet group for Cafe" \
> --subnet-ids subnet-0707b61b9b3f29611 subnet-03e949f623dc95cbc \
> --tags "Key=Name,Value= CafeDatabaseSubnetGroup"
{
    "DBSubnetGroup": {
        "Subnets": [
            {
                "SubnetStatus": "Active",
                "SubnetIdentifier": "subnet-0707b61b9b3f29611",
                "SubnetOutpost": {},
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2a"
                }
            },
            {
                "SubnetStatus": "Active",
                "SubnetIdentifier": "subnet-03e949f623dc95cbc",
                "SubnetOutpost": {},
                "SubnetAvailabilityZone": {
                    "Name": "us-west-2b"
                }
            }
        ],
        "VpcId": "vpc-0bdd6f3c88e6383d0",
        "DBSubnetGroupDescription": "DB subnet group for Cafe",
        "SubnetGroupStatus": "Complete",
        "DBSubnetGroupArn": "arn:aws:rds:us-west-2:525969271926:subgrp:cafedb subnet group",
        "DBSubnetGroupName": "cafedb subnet group"
    }
}
```

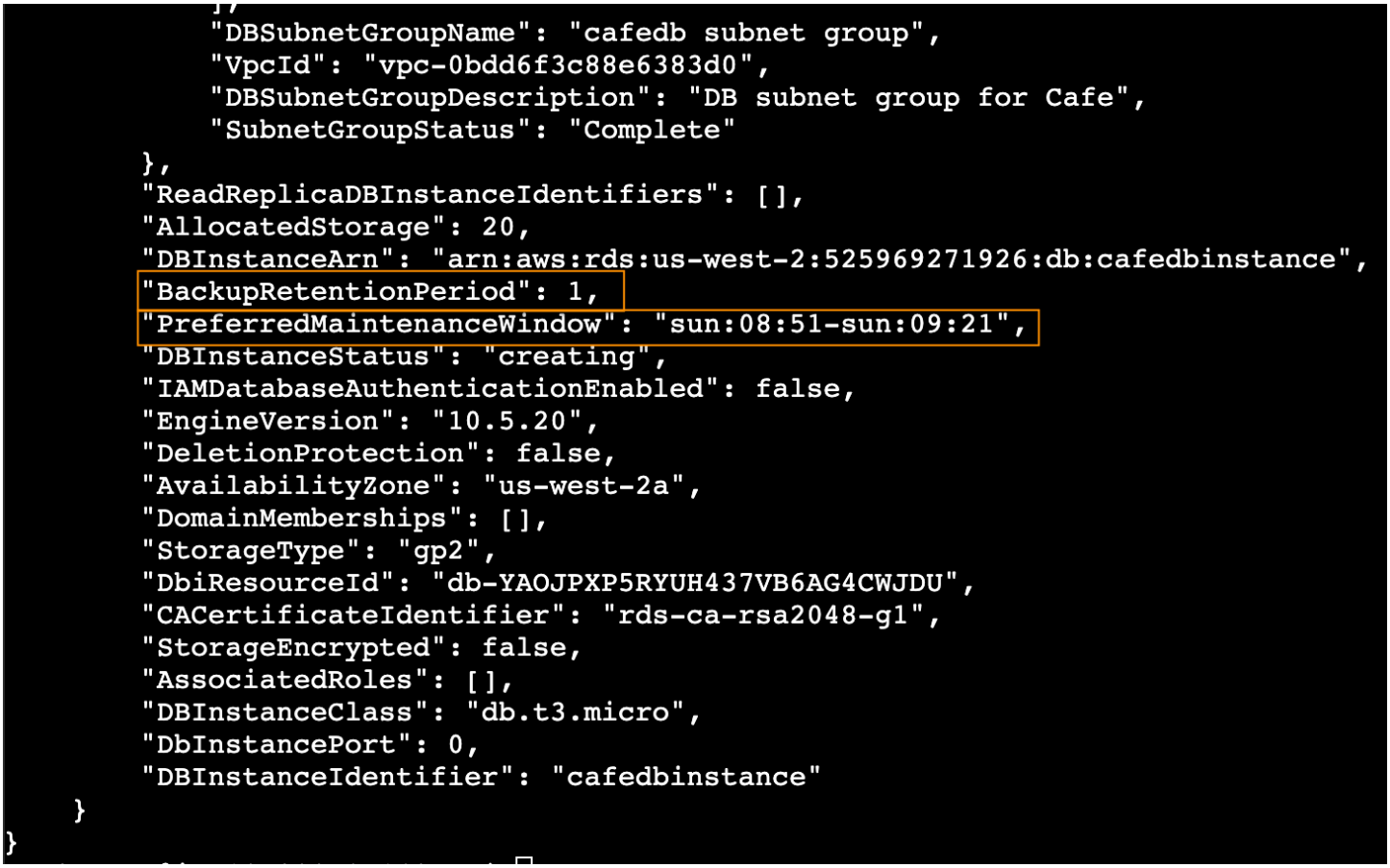
- **Creating the Amazon RDS MariaDB instance using AWS CLI:**

```
[ec2-user@ip-                ]$ aws rds create-db-instance \
> --db-instance-identifier CafeDBInstance \
> --engine mariadb \
> --engine-version 10.5.20 \
> --db-instance-class db.t3.micro \
> --allocated-storage 20 \
> --availability-zone us-west-2a \
> --db-subnet-group-name "CafeDB Subnet Group" \
> --vpc-security-group-ids sg-03d1b5d5a80e155e0 \
> --no-publicly-accessible \
> --master-username                    -password
```

This command immediately returns some information about the database, but the database might take up to 10 minutes to become available.

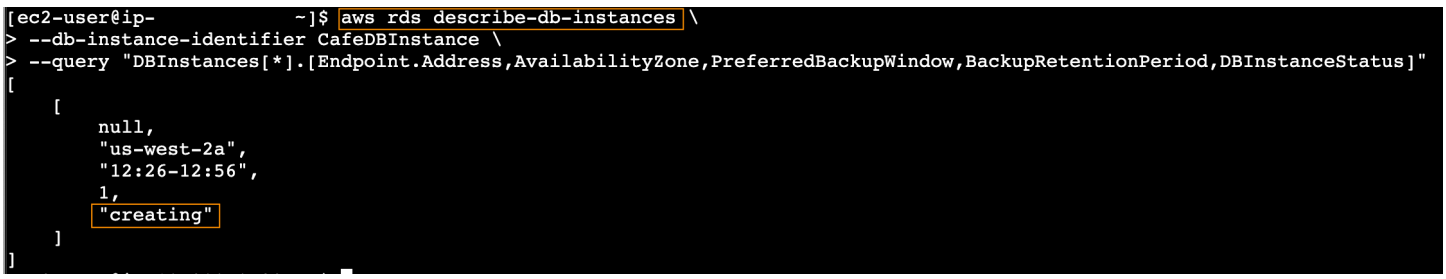
Note the value of '*1*' for the backup retention period, which indicates that, by default, daily backups are retained for only 1 day.

The preferred backup window is set to a 30 minute time interval by default. We can modify these settings to match our desired backup policy.



Monitoring the status of the database:

The status attribute initially shows the value '*creating*', then changes to '*modifying*', after that '*backing-up*' and finally '*available*'.



When the status shows '*available*', it returns the value for the **endpoint address** following the next format: *cafedbinstance.xxxxxxx.us-west-2.rds.amazonaws.com.*

```
[ec2-user@ip-                ~]$ aws rds describe-db-instances --db-instance-identifier CafeDBInstance --query "DBInstances[*].[Endpoint.Address,AvailabilityZone,PreferredBackupWindow,BackupRetentionPeriod,DBInstanceStatus]"
[
    [
        "cafedbinstance.cmnnirnun7ho.us-west-2.rds.amazonaws.com",
        "us-west-2a",
        "10:15-10:45",
        1,
        "available"
    ]
]
```

3 . Migrating Application Data to the Amazon RDS instance

I migrated the data from the existing local database to the newly created Amazon RDS database.

- **Connecting to the CafeInstance by using EC2 Instance Connect:**

I used **EC2 Instance Connect** to access the CafeInstance (EC2 instance):

  - Access the **EC2 Management Console** and navigate to **Instances.**
  - Select the CafeInstance instance.
  - Click **Connect**, then choose the EC2 Instance Connect tab.
  - Click connect to establish the session.

- **Creating a backup of the local database:**

Using *mysqldump* I generated a SQL statement in a file named '*cafedb-backup.sql*', which can be run to reproduce the schema and data of the original cafe_db database.

```
[ec2-user@ip-                ~]$ mysqldump --user=     --password='                \
> --databases cafe_db --add-drop-database > cafedb-backup.sql
```

Reviewing the *cafedb-backup.sql* file generated using **Nano:**

```
 GNU nano 2.9.8                                  cafedb-backup.sql

-- MySQL dump 10.19  Distrib 10.2.38-MariaDB, for Linux (x86_64)
--
-- Host: localhost    Database: cafe_db
-- ------------------------------------------------------
-- Server version       10.2.38-MariaDB

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Current Database: `cafe_db`
--

/*!40000 DROP DATABASE IF EXISTS `cafe_db`*/;

CREATE DATABASE /*!32312 IF NOT EXISTS*/ `cafe_db` /*!40100 DEFAULT CHARACTER SET latin1 */;

USE `cafe_db`;

--
-- Table structure for table `order`
--

DROP TABLE IF EXISTS `order`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `order` (
  `order_number` int(5) NOT NULL AUTO_INCREMENT,
  `order_date_time` timestamp NOT NULL DEFAULT current_timestamp(),
  `amount` decimal(10,2) NOT NULL DEFAULT 0.00,
  PRIMARY KEY (`order_number`)
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=latin1;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `order`
--

LOCK TABLES `order` WRITE;
/*!40000 ALTER TABLE `order` DISABLE KEYS */;
INSERT INTO `order` VALUES (1,'2025-03-13 07:55:48',21.50);
/*!40000 ALTER TABLE `order` ENABLE KEYS */;

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos  M-U Undo  M-A Mark Text  M-] To Bracket  M-▲ Previous  ^B Back  ^◀ Prev Word  ^A Home
^X Exit      ^R Read File  ^\ Replace   ^U Uncut Text ^T To Spell ^_ Go To Line M-E Redo  M-6 Copy Text M-W WhereIs Next M-▼ Next  ^F Forward ^▶ Next Word ^E End
```

- **Restoring the backup to the Amazon RDS database using the mySQL command:**

This command creates a MySQL connection to the Amazon RDS instance (specified with the RDS database Endpoint Address) and runs the SQL statement in the file '*cafedb-backup.sql*'.

```
[ec2-user@ip- ~]$ mysql --user=  --password='  ' \
> --host=cafedbinstance.  .us-west-2.rds.amazonaws.com \
> < cafedb-backup.sql
```

- **Testing the data migration**:

I verified that the cafe_db was successfully created and populated in the Amazon RDS instance.

  - Open an interactive MySQL session to the instance.

```
[ec2-user@ip-  ~]$ mysql --user=  --password='  ' \
> --host=cafedbinstance.  .us-west-2.rds.amazonaws.com
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 55
Server version: 10.5.20-MariaDB-log managed by https://aws.amazon.com/rds/

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

  - Retrieving the data in the '*product*' table of the cafe_db database:

```
MariaDB [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| cafe_db            |
| information_schema |
| innodb             |
| mysql              |
| performance_schema |
+--------------------+
5 rows in set (0.01 sec)
```

```
MariaDB [(none)]> USE cafe_db;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [cafe_db]>
```

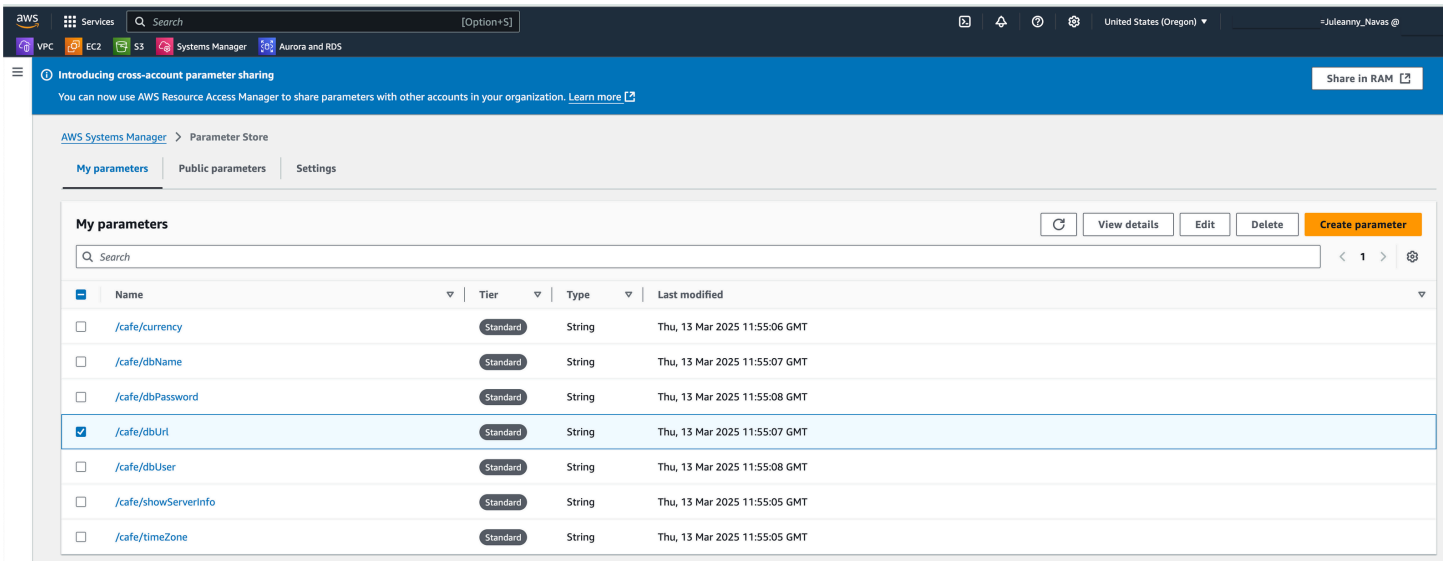
```
MariaDB [cafe_db]> SELECT *
    -> FROM product;
+----+--------------------------+-----------------------------------------------------------+-------+---------------+-----------------------------------+
| id | product_name             | description                                               | price | product_group | image_url                         |
+----+--------------------------+-----------------------------------------------------------+-------+---------------+-----------------------------------+
|  1 | Croissant                | Fresh, buttery and fluffy... Simply delicious!            |  1.50 |             1 | images/Croissants.jpg             |
|  2 | Donut                    | We have more than half-a-dozen flavors!                   |  1.00 |             1 | images/Donuts.jpg                 |
|  3 | Chocolate Chip Cookie    | Made with Swiss chocolate with a touch of Madagascar vanilla |  2.50 |             1 | images/Chocolate-Chip-Cookies.jpg |
|  4 | Muffin                   | Banana bread, blueberry, cranberry or apple               |  3.00 |             1 | images/Muffins.jpg                |
|  5 | Strawberry Blueberry Tart | Bursting with the taste and aroma of fresh fruit          |  3.50 |             1 | images/Strawberry-Blueberry-Tarts.jpg |
|  6 | Strawberry Tart          | Made with fresh ripe strawberries and a delicious whipped cream |  3.50 |             1 | images/Strawberry-Tarts.jpg       |
|  7 | Coffee                   | Freshly-ground black or blended Columbian coffee          |  3.00 |             2 | images/Coffee.jpg                 |
|  8 | Hot Chocolate            | Rich and creamy, and made with real chocolate             |  3.00 |             2 | images/Cup-of-Hot-Chocolate.jpg   |
|  9 | Latte                    | Offered hot or cold and in various delicious flavors      |  3.50 |             2 | images/Latte.jpg                  |
+----+--------------------------+-----------------------------------------------------------+-------+---------------+-----------------------------------+
9 rows in set (0.00 sec)
```

4 . Configuring the Website to use the Amazon RDS instance

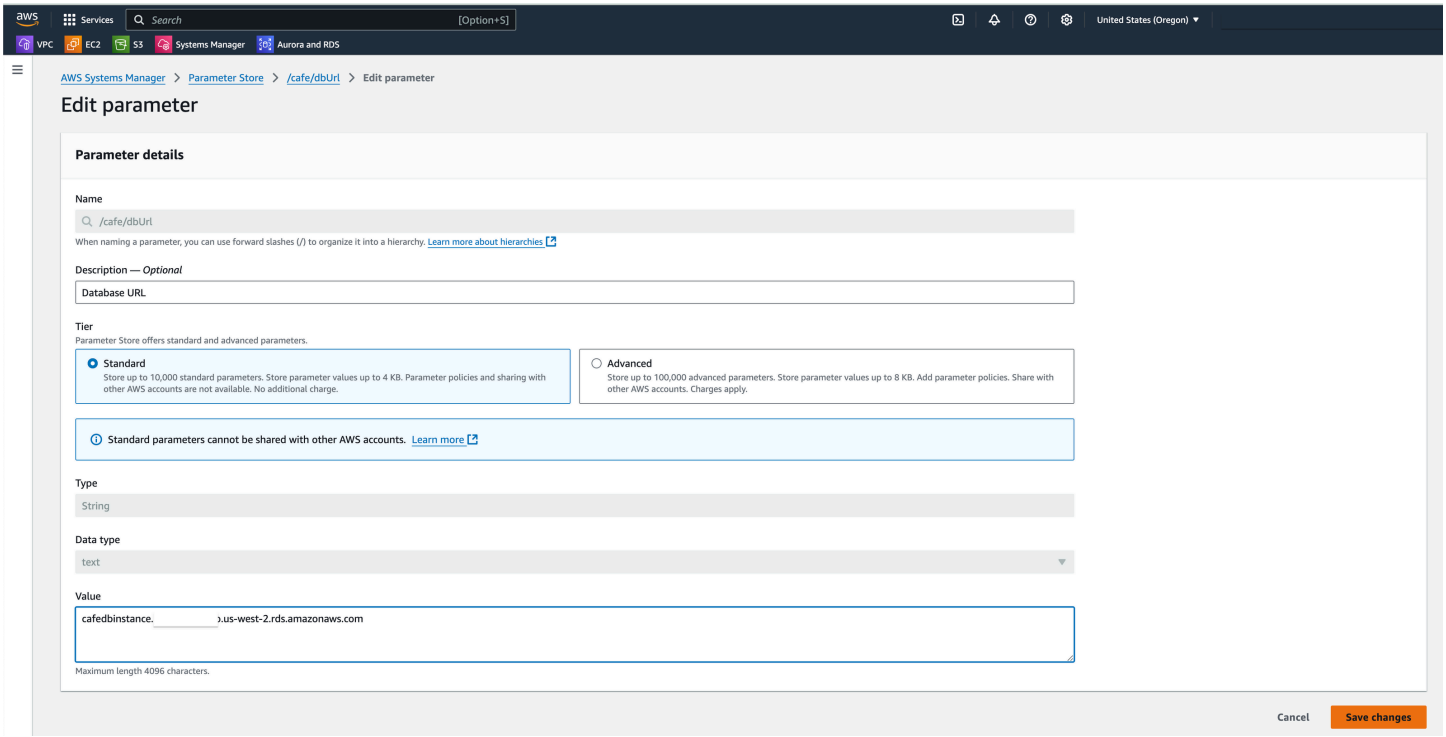
For the purpose of this practice I consider that the database connection information was externalized as parameters in **Parameter Store** (a capability of **AWS System Manager**).

In order to configure the Website, I changed the database URL parameter of the Cafe Application to point to the endpoint address of the RDS instance, following these steps:

- On the **AWS Management Console** went to **AWS System Manager.** In the left pane chose **Parameter Store.** From 'My Parameters' list chose */ cafe / dbURL*. Edit.

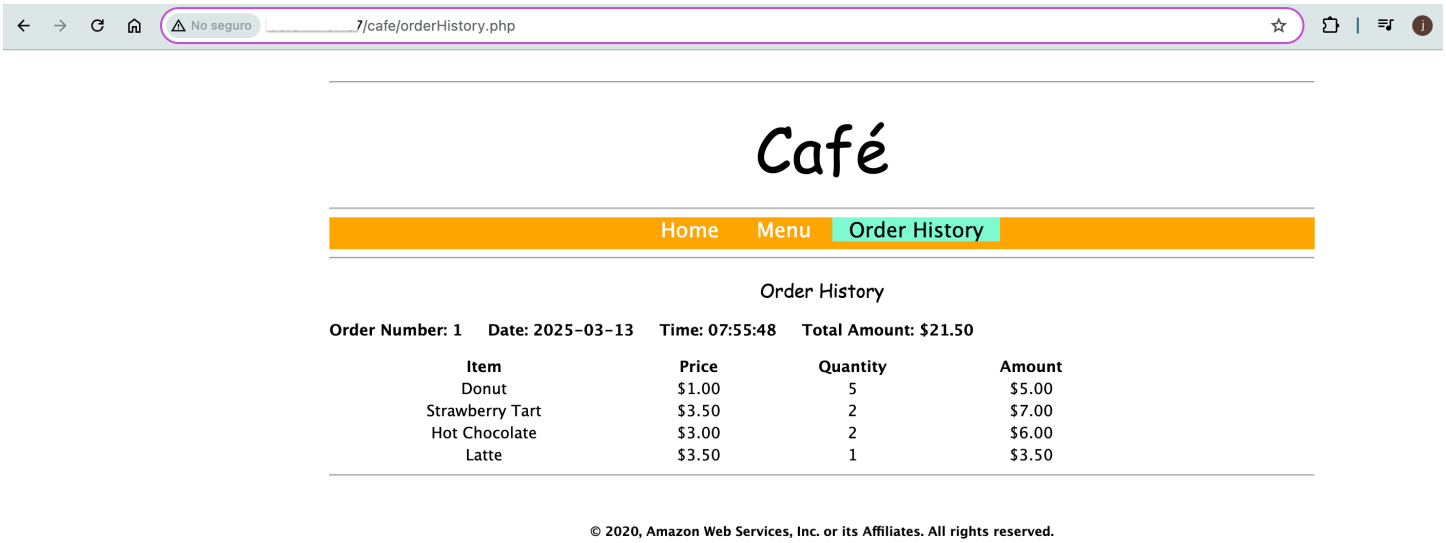


- Edit Parameter details, changing value for RDS database Endpoint Address. Save changes.



- **Testing the Website:**

I confirmed the website could access the new database correctly by pasting the CafeInstance URL into a new browser window, selecting **'Order History'** and comparing the displayed data with the records from step 1, before the database migration.



5 . Monitoring the Amazon RDS database

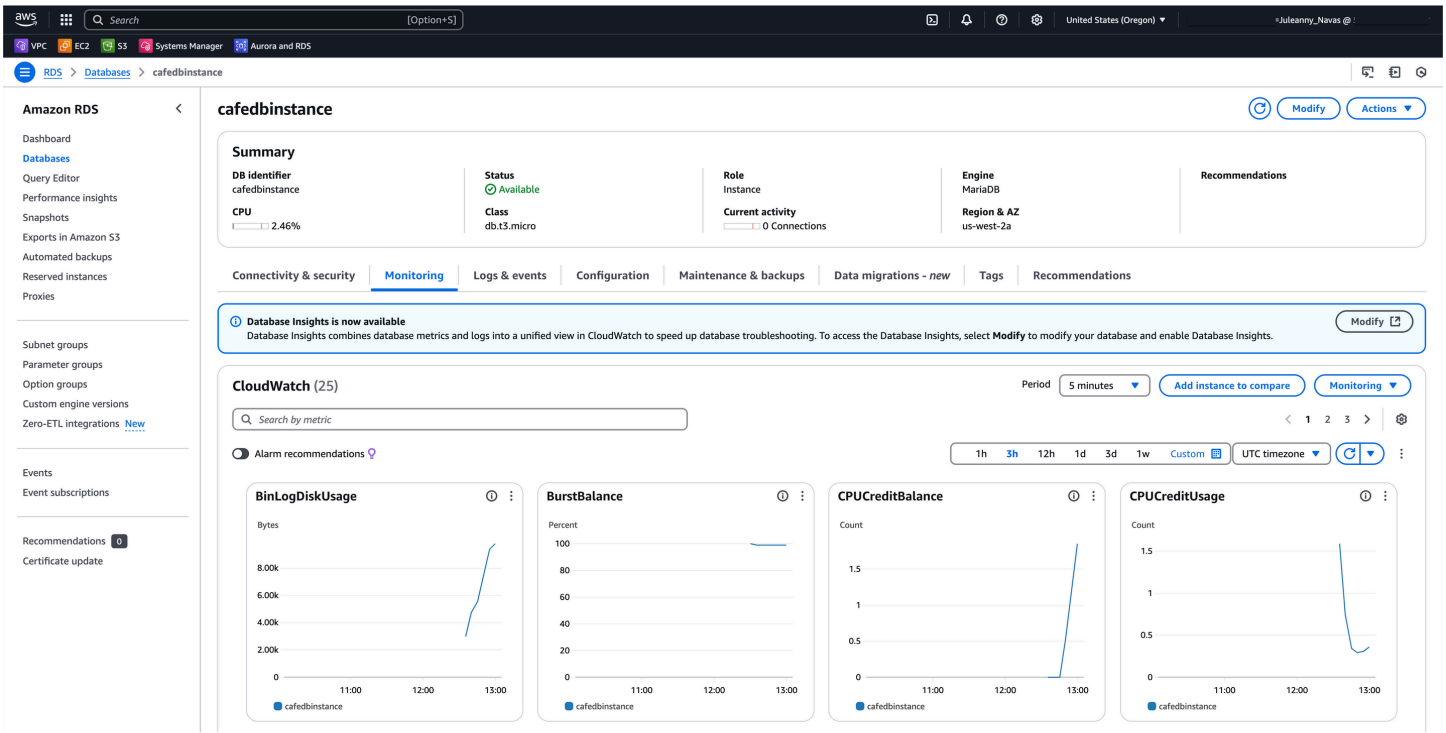
Since Amazon RDS automatically sends metrics to **CloudWatch** every minute for each active database, I identified some of these metrics through the **Amazon RDS console**.

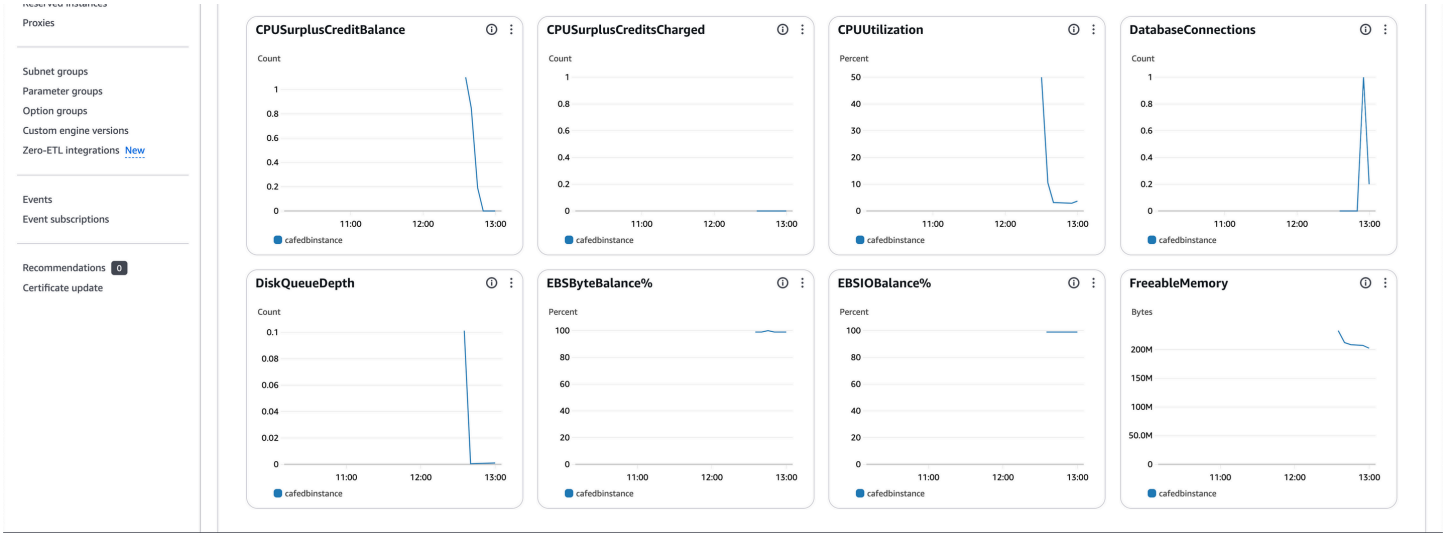
Steps taken:

1. Opened the **AWS Management Console** and navigated to **RDS.**
2. Selected **Databases** and chose **cafedbinstance** from the list.
3. Clicked on the **Monitoring** tab to view key database instance metrics.

The list of metric include the following:

- **CPUUtilization:** The percent of CPU utilization.
- **DatabaseConnections:** The number of database connections in use.
- **FreeStorageSpace:** The amount of available storage space.
- **FreableMemory:** The amount of memory (RAM) available on the Amazon RDS instance.
- **WriteIOPS:** The average number of disk write I/O operations per second.
- **ReadIOPS:** The average number of disk read I/O operations per second.





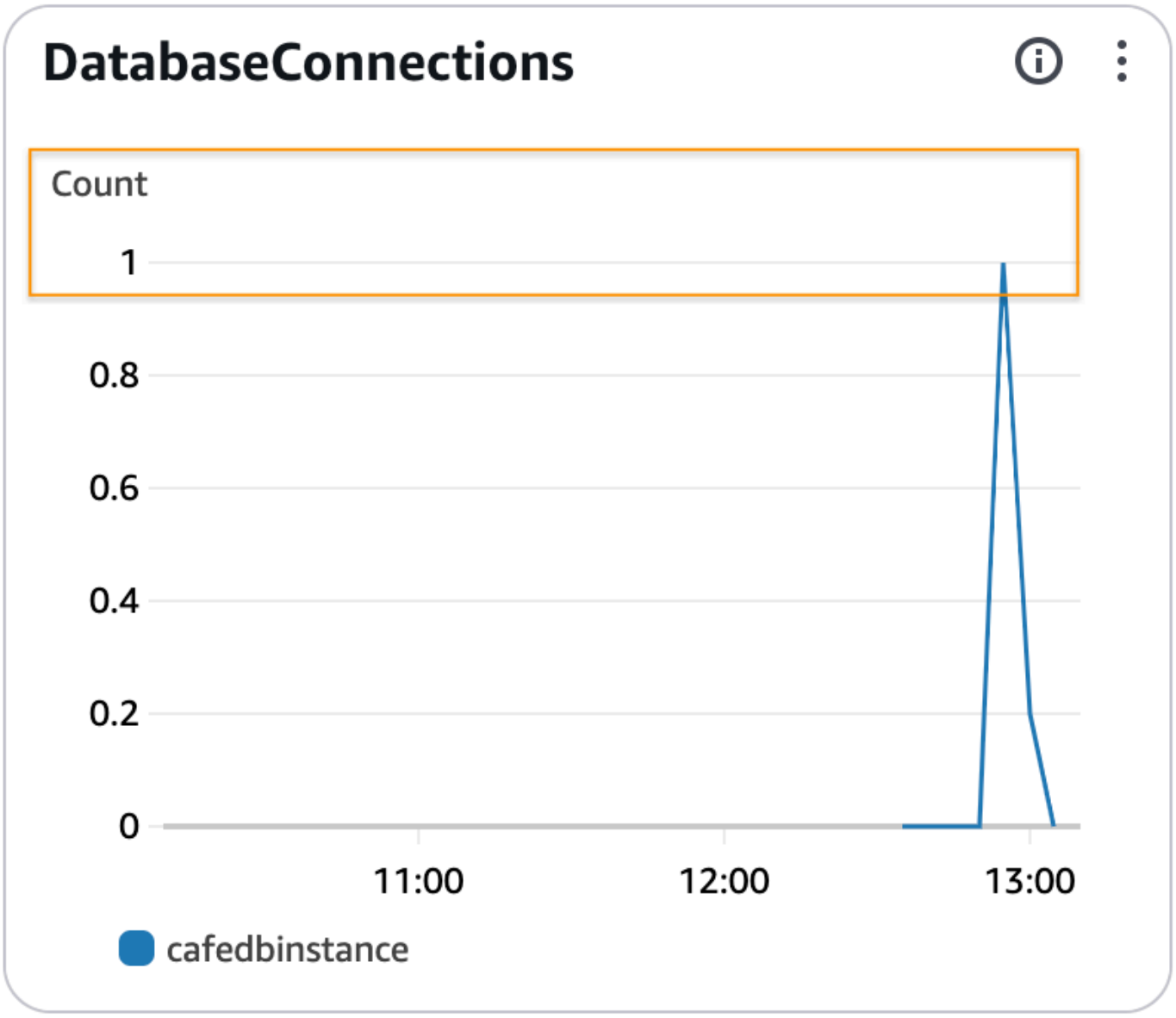
- **Monitoring the DatabaseConnections metric while creating a connection to the database from the CafeInstance:**
  - Opening a SQL session to the RDS cafe_db instance, in the CafeInstance terminal:

```
[ec2-user@ip-10-200-0-158 ~]$ mysql --user=     --password='          ' \
> --host=cafedbinstance.cmnnirnun7ho.us-west-2.rds.amazonaws.com \
> cafe_db
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 81
Server version: 10.5.20-MariaDB-log managed by https://aws.amazon.com/rds/

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [cafe_db]> select * from product;
```

  - Retrieve the data in the '*product*' table.

```
MariaDB [cafe_db]> SELECT *
    -> FROM product;
+----+--------------------------+-----------------------------------------------------------------+-------+---------------+-----------------------------------+
| id | product_name             | description                                                     | price | product_group | image_url                         |
+----+--------------------------+-----------------------------------------------------------------+-------+---------------+-----------------------------------+
|  1 | Croissant                | Fresh, buttery and fluffy... Simply delicious!                  |  1.50 |             1 | images/Croissants.jpg             |
|  2 | Donut                    | We have more than half-a-dozen flavors!                         |  1.00 |             1 | images/Donuts.jpg                 |
|  3 | Chocolate Chip Cookie    | Made with Swiss chocolate with a touch of Madagascar vanilla    |  2.50 |             1 | images/Chocolate-Chip-Cookies.jpg |
|  4 | Muffin                   | Banana bread, blueberry, cranberry or apple                     |  3.00 |             1 | images/Muffins.jpg                |
|  5 | Strawberry Blueberry Tart | Bursting with the taste and aroma of fresh fruit               |  3.50 |             1 | images/Strawberry-Blueberry-Tarts.jpg |
|  6 | Strawberry Tart          | Made with fresh ripe strawberries and a delicious whipped cream |  3.50 |             1 | images/Strawberry-Tarts.jpg       |
|  7 | Coffee                   | Freshly-ground black or blended Columbian coffee                |  3.00 |             2 | images/Coffee.jpg                 |
|  8 | Hot Chocolate            | Rich and creamy, and made with real chocolate                   |  3.00 |             2 | images/Cup-of-Hot-Chocolate.jpg   |
|  9 | Latte                    | Offered hot or cold and in various delicious flavors            |  3.50 |             2 | images/Latte.jpg                  |
+----+--------------------------+-----------------------------------------------------------------+-------+---------------+-----------------------------------+
9 rows in set (0.00 sec)
```

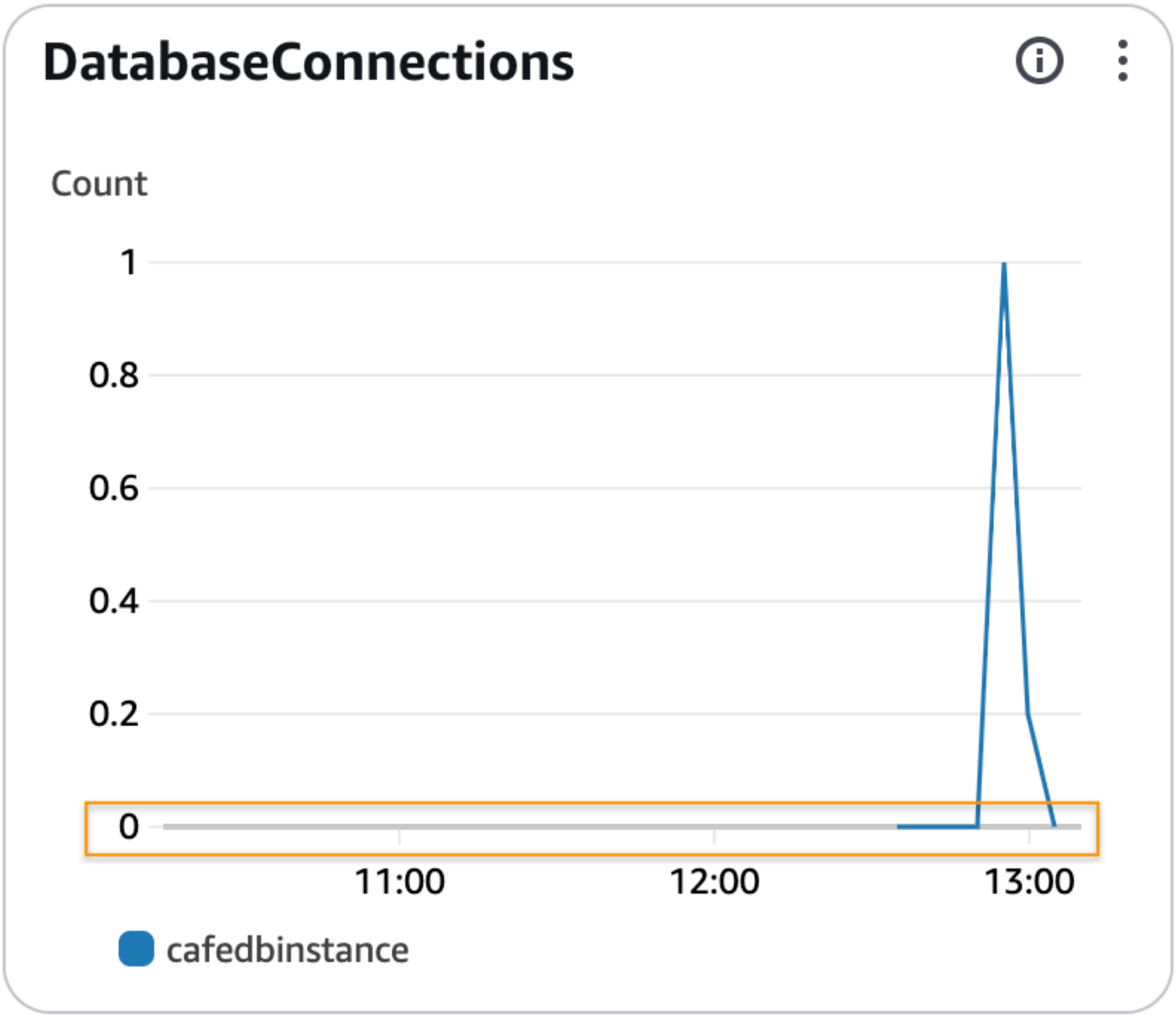
  - In the Amazon RDS console, chose the **DatabaseConnections** graph, it showed a line indicating that **1** connection was used. This connection was established by the SQL session from the CafeInstance.



- Closing the connection from the SQL session, in the CafeInstance terminal.

```
MariaDB [cafe_db]> exit
Bye
[ec2-user@ip-                 ~]$
```

- Checking the **DatabaseConnections** after a minute (Choosing refresh). The graph now shows that the number of connections in use is **0**.


DatabaseConnections
Count
1
0.8
0.6
0.4
0.2
0
11:00
12:00
13:00
cafedbinstance

Conclusions & Lessons Learned

Migrating a **MariaDB database** from an **EC2 instance to Amazon RDS** significantly improves **availability, scalability, and operational efficiency**. My key takeaways include:

1. <u>High Availability with Amazon RDS</u>

- **Multi-AZ Deployments**: RDS automatically maintains a standby replica in another **Availability Zone**, ensuring seamless **failover** with minimal downtime.
- **Automated Backups & Snapshots**: RDS offers **point-in-time recovery**, eliminating the need for manual backup management.
- **Built-in Failover Handling**: Unlike EC2, where failover setup is manual, RDS automates this process.

2. <u>Automated Scaling & Performance Improvements</u>

- **Vertical Scaling**: RDS allows **easy instance type upgrades** without OS-level configurations.
- **Horizontal Scaling with Read Replicas:** Distributes read traffic efficiently, reducing database load.
- **Storage Auto Scaling:** Automatically increases storage to prevent capacity-related outages.

3. <u>Reduced Operational Overhead</u>

- **Fully Managed Service:** AWS handles **maintenance, patching, and updates**, unlike EC2, where these tasks must be manually managed.
- **Integrated Monitoring** (**Amazon CloudWatch):** provides real-time performance insights and alerts without additional setup.

4. <u>Security & Compliance</u>

- **Automated Security Patches**: RDS applies patches automatically, improving security.
- **Fine-Grained Access Control: AWS IAM integration** simplifies authentication and permissions management.
- **Encryption at Rest & In Transit**: RDS supports **AWS KMS encryption** and **SSL/TLS** for secure data transmission.

<u>Final Thoughts</u>

Migrating to **Amazon RDS** significantly reduces administrative burden, enhances security, and improves database performance. Leveraging **AWS best practices**, such as **staging migrations, monitoring with CloudWatch, and parameter management with AWS Systems Manager**, ensures a **smooth transition** with **minimal downtime.**