

ExtraCredit – RentTrack Integration

BuildCreditModule AWS and Heroku Infra Setup

Table of Contents

[1. High-level Infra Setup 3](#_Toc25614932)

[a. Non-Prod and Prod 3](#_Toc25614933)

[2. Environment: Dev 4](#_Toc25614934)

[a. Database 4](#_Toc25614935)

[b. ActiveMQ 9](#_Toc25614936)

[c. Heroku Shield Private Space 11](#_Toc25614937)

[3. Environment: Test 12](#_Toc25614938)

[a. Database 12](#_Toc25614939)

[b. ActiveMQ 13](#_Toc25614940)

[c. Heroku Shield Private Space 13](#_Toc25614941)

[4. Environment: Stage 13](#_Toc25614942)

[a. Database 13](#_Toc25614943)

[b. ActiveMQ 14](#_Toc25614944)

[c. Heroku Shield Private Space 17](#_Toc25614945)

[5. Environment: Production 17](#_Toc25614946)

[a. Database 17](#_Toc25614947)

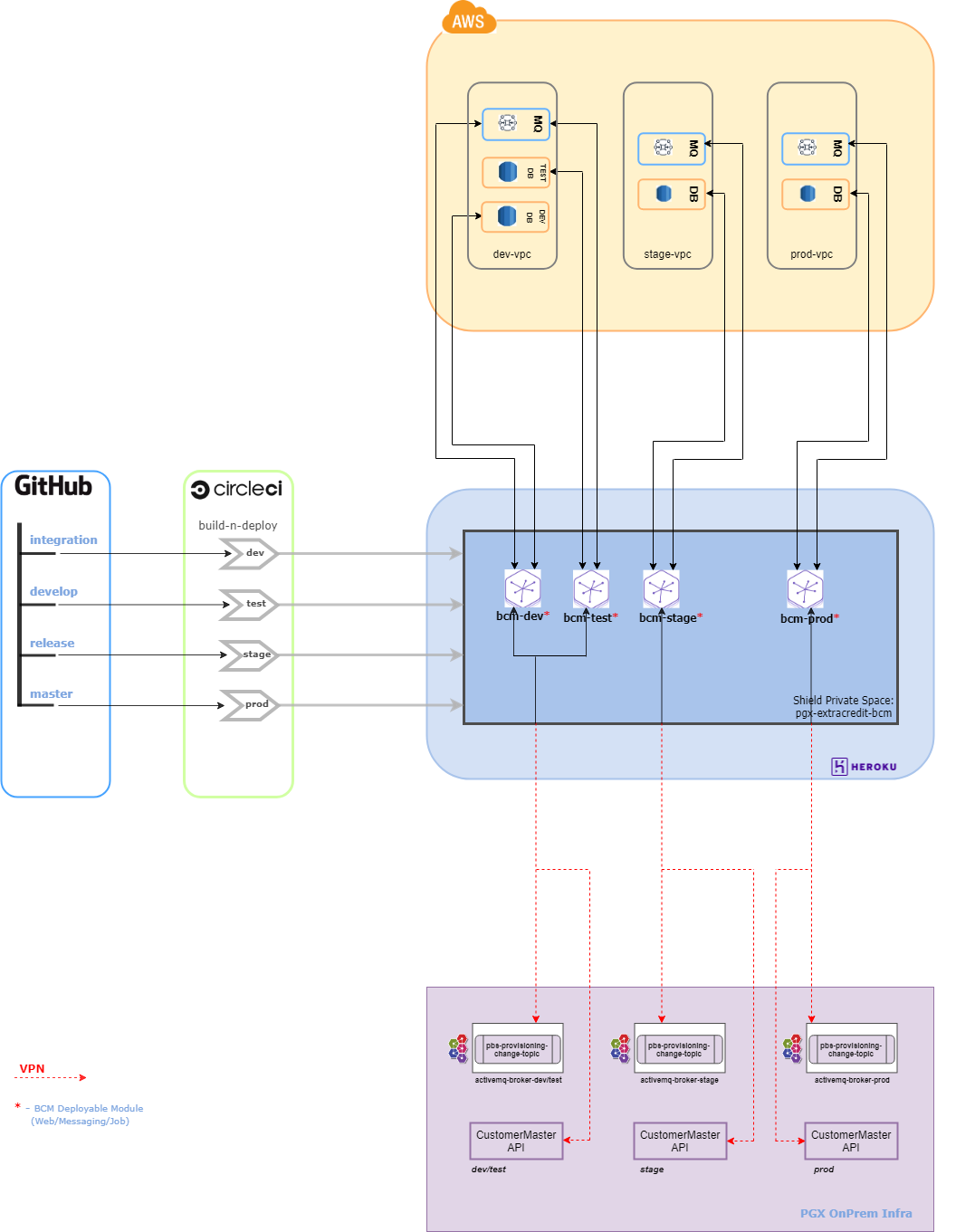
[b. ActiveMQ 18](#_Toc25614948)

[c. Heroku Shield Private Space 21](#_Toc25614949)

# High-level Infra Setup

## Non-Prod and Prod

The diagram given below gives a high-level view of the AWS and Heroku Infra setup in Non-Prod as well as Prod. Connectivity between AWS VPC and Heroku Shield Private Space is established through VPC Peering. Dev and Test Database instances are provisioned in the ‘dev-vpc’, and the existing MQ Broker in ‘dev-vpc’ is used for both Dev and Test region Queue. Stage and Prod database instances are provisioned in ‘stage-vpc’ and ‘prod-vpc’ respectively. The existing stage and prod MQ brokers are used for hosting BCM’s Stage and Prod regions Queues.



# Environment: Dev

## Database

For BCM, we would be using the Amazon Aurora DB with PostgreSQL compatible version

#### Environment Details

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| db cluster name | Endpoints | DB name | Engine version | Multi AZ deployment | VPC |
| dbclusteridentifier-extracresit-bcm-dev-data01 | [Writer]  dbclusteridentifier-extracredit-bcm-dev-data01.cluster-czqcrnu8rq1a.us-west-2.rds.amazonaws.com  [Reader] dbclusteridentifier-extracredit-bcm-dev-data01.cluster-ro-czqcrnu8rq1a.us-west-2.rds.amazonaws.com | BuildCredit | 10.7 | No | extracredit-dev-vpc-vpc (vpc-0ccd8c86dea26ace2) |

We have created a role ‘**bcm’** which is having the read write permissions, also we have created a role ‘**bcm-service-user’** as a common login for the application to connect to the database. **This is applicable in all the environments.**

**When any database objects (tables, etc.) are created in Postgres DB, the user who created it becomes the Owner of that object by default. To make sure the application user (bcm-service-user) has access to the object, whoever creates any new object should immediately change the owner of that object to the role: ‘bcm’. This ensures that the application user (bcm-service-user) and others who belong to the ‘bcm’ role have access to the object.**

For example, to change the owner of a table, we can use the following syntax:

**ALTER TABLE** <schema\_name>.<table\_name> **OWNER TO “**<role\_name>”;

We have created a read-only role ‘**bcm-readonly**’ for mapping read-only access to users who don’t required write permissions. In order to make the role read-only we have followed the below steps. NOTE: The Master User created when setting up the database (through Cloudformation template) are used for executing the below steps.

**Steps to create a user, role and map the role to the user in PostgreSQL :**

* 1. **To create a new user in PostgreSQL:**

CREATE USER username WITH PASSWORD 'your\_password';

* 1. **To create a new role**

CREATE ROLE <role\_name> WITH NOLOGIN;

* 1. **To map the role to a user**

GRANT <role\_name> TO username;

**Steps to be followed for setting up a ‘read-only’ role:**

* 1. **GRANT the CONNECT access:**

GRANT CONNECT ON DATABASE <database\_name> TO <role >;

* 1. **Then GRANT USAGE on schema:**

GRANT USAGE ON SCHEMA <schema\_name> TO <role >;

* 1. **GRANT SELECT**

GRANT SELECT ON ALL TABLES IN SCHEMA <schema\_name> TO <role >;

* 1. **To grant access to the new table in the future automatically, alter default:**

ALTER DEFAULT PRIVILEGES IN SCHEMA <schema\_name>

GRANT SELECT ON TABLES TO <role >;

#### Configuration Steps

AWS CloudFormation provides a common language for you to describe and provision all the infrastructure resources in your cloud environment. CloudFormation allows you to use programming languages or a simple text file to model and provision, in an automated and secure manner, all the resources needed for your applications across all regions and accounts.

GitHub Location where the Cloudformation Templates for BCM are stored:

URL: <https://github.com/progrexion/extracredit-bcm-services.git>

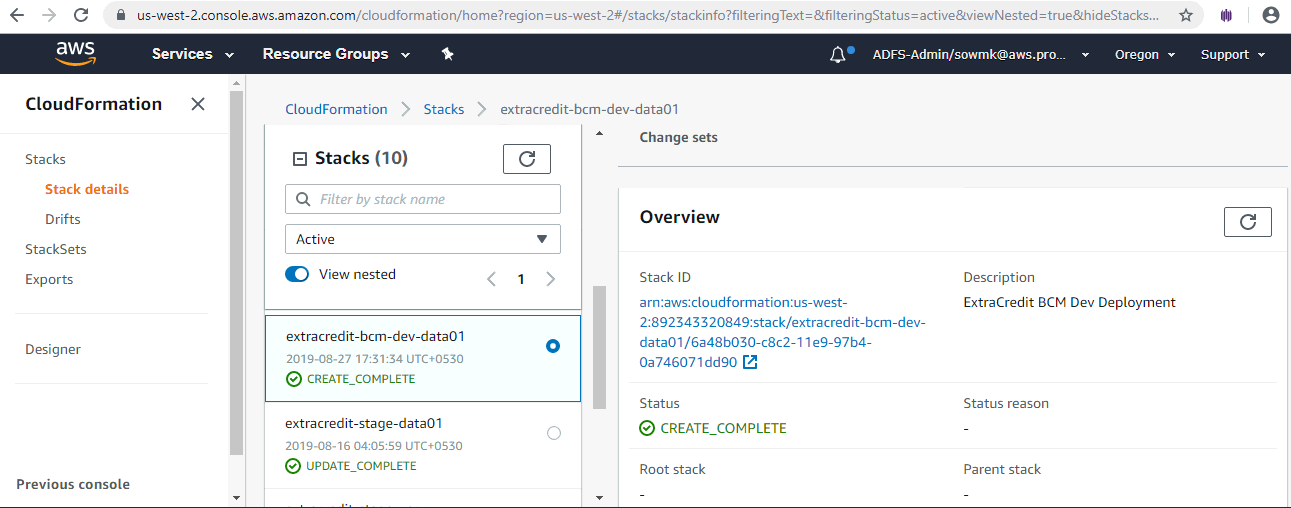
Path: **./docs/cloudformation-templates**

The database in dev environment is created through CloudFormation, following the existing templates which was used for Rewards module (template name: ExtraCredit BCM Dev Deployment). Below provided are the step by step details for the same:

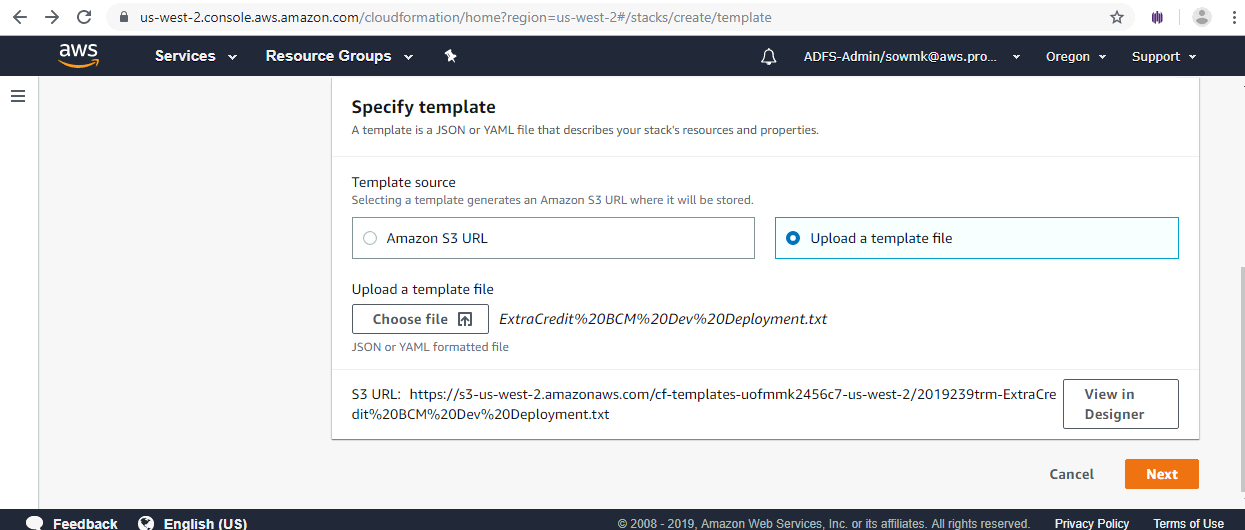
Step1:

Login to the AWS console > Go to CloudFormation > Click on Stacks > Click on Create Stack

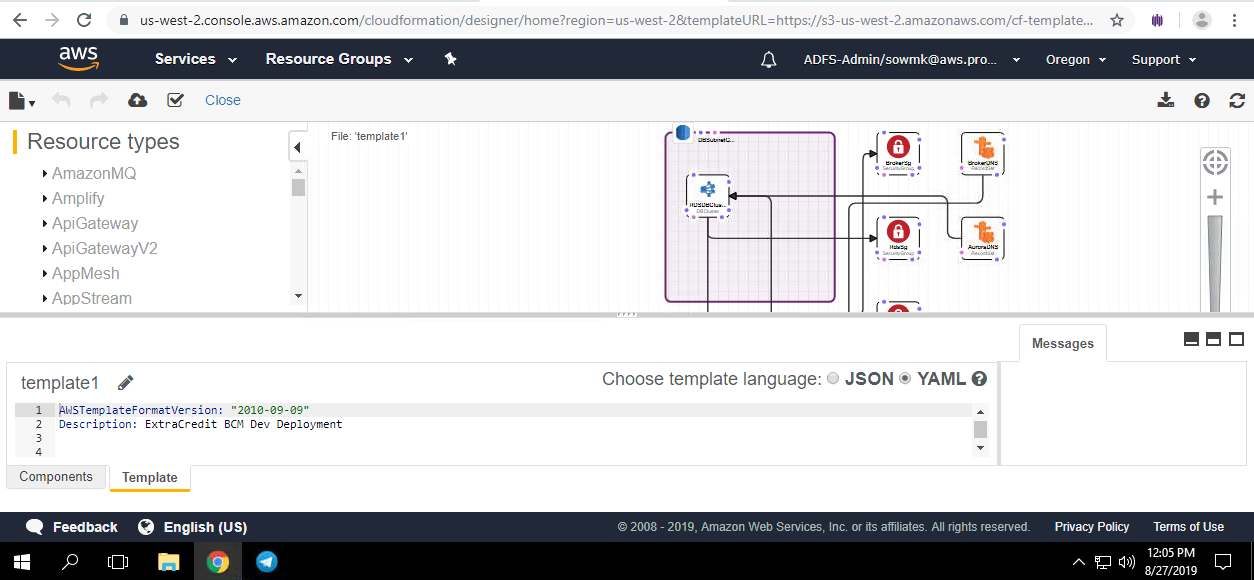
Click on “Upload a template file”



Step2: Upload the template file. The template is written in JSON format. For dev environment we are going with single instance deployment with no read replica, and all the read and write operations will be done on the primary database instance.

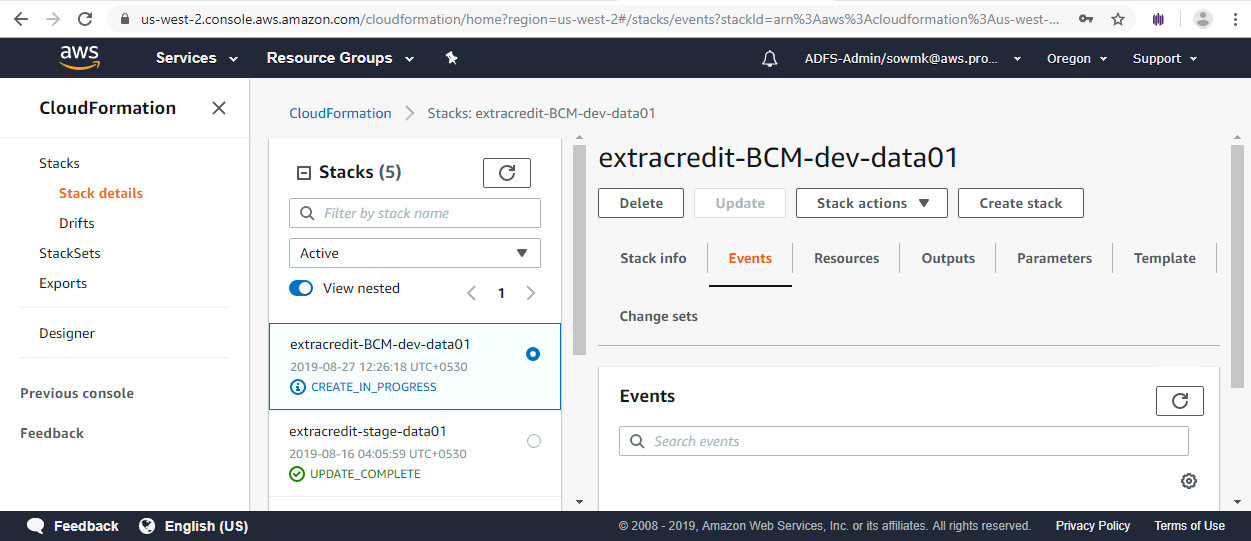


Step 3: After uploading the template, please validate the template by clicking the validate button (“Check mark” icon on the top left). If the template is valid click on Create Stack. If the template is invalid, check the errors and fix them.

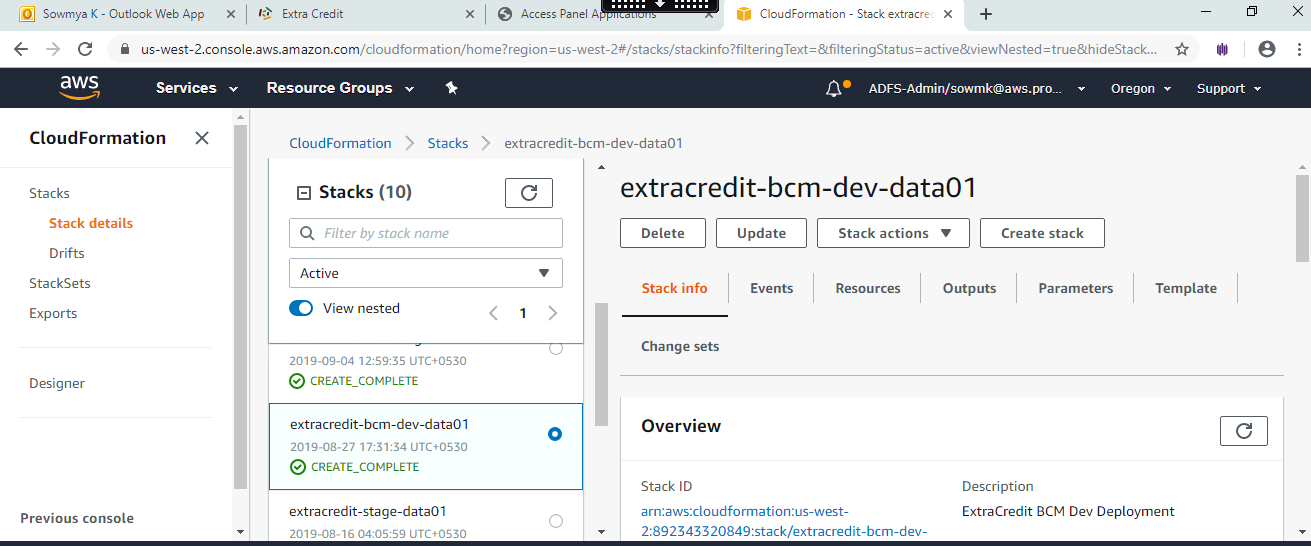


Step 4: Once the template is valid we will be providing parameters such as database name, MasterUsername and MasterUserPassword.

After entering the parameters, review the details and click on create stack. You can see the status CREATE\_IN\_PROGRESS, meanwhile we can see the events taking place - that is, resource creation happening one after the other.



Once the creation is complete, the status can be seen as CREATE\_COMPLETE. In case there are any errors, the stack creation will be rolled back, and we need to fix the errors and create the stack again.



## ActiveMQ

Amazon MQ is a managed message broker service for Apache ActiveMQ that makes it easy to set up and operate message brokers in the cloud. Message brokers allow different software systems–often using different programming languages, and on different platforms–to communicate and exchange information. Amazon MQ reduces your operational load by managing the provisioning, setup, and maintenance of ActiveMQ, a popular open-source message broker.

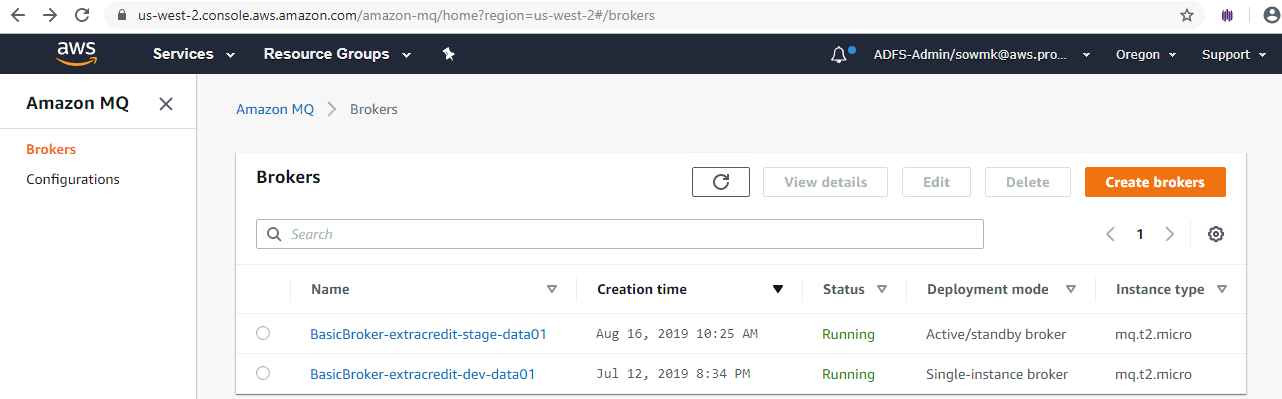
ActiveMQ Brokers have been created already for Dev and Stage (as part of Rewards Module’s setup), and BCM would be using those brokers to host the relevant Queue(s). There is no separate broker for Test environment - we would be using the one in Dev.

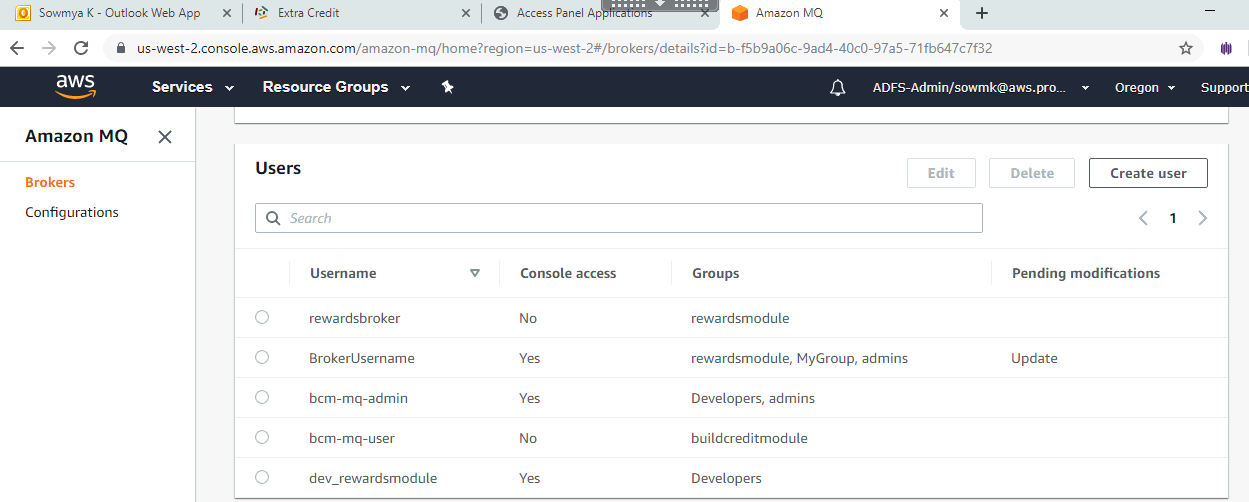
#### Environment Details

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Broker name | Broker instance type | Deployment mode | Active MQ Web console | Endpoint | Users created | Groups created |
| BasicBroker-extracredit-dev-data01 | mq.t2.micro | Single-instance broker | https:///b-f5b9a06c-9ad4-40c0-97a5-71fb647cf32-1.mq.us-west-2amazon.com:8162 | ssl://b-f5b9a06c-9ad4-40c0-97a5-71fb647cf32-1.mq.us-west-2amazon.com:61617 | bcm-mq-admin  bcm-mq-user | Developers,  Admins,buildcreditmodule |

#### Configuration Steps

The dev and test uses the single instance broker.

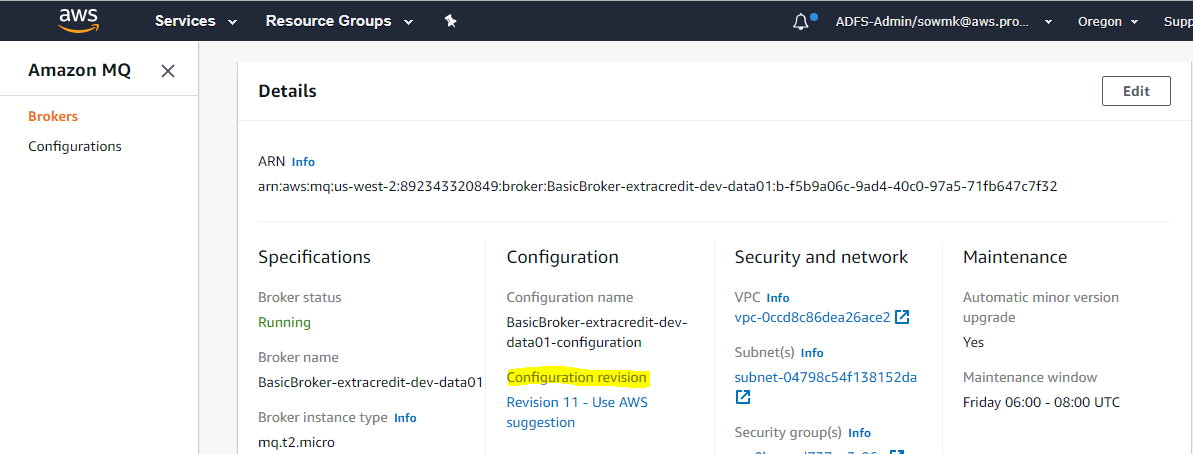




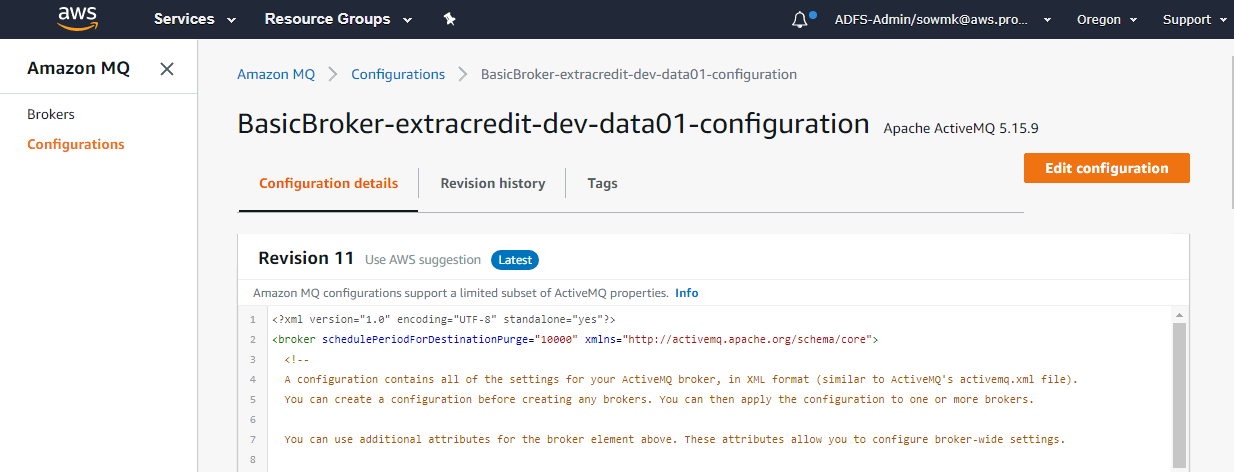
The Authorization plugin info has been modified to add the groups related to bcm module.



To change the Authorization plugin info navigate to Amazon MQ in the AWS console, select the relevant broker for dev(BasicBroker-extracredit-dev-data01) and in the details page under configuration revision click on the revision link



Click on edit configuration as shown in the below screen and make the necessary changes and save the changes.



Once the changes are saved, Edit the Broker, choose the Configuration Revision to apply, and Schedule the modifications. Opt for rebooting the broker immediately to get the changes take effect without waiting for the maintenance window.

## Heroku Shield Private Space

#### Environment Details

|  |  |  |  |
| --- | --- | --- | --- |
| Team Name | Shield space name | Region | CIDR range |
| progrexion | pgx-extracredit-bcm | Oregon | 10.99.0.0/16 |

#### Configuration Steps

Command used to create the Shield Private Space:

**> heroku spaces:create --cidr 10.99.0.0/16 --space pgx-extracredit-bcm --shield --team progrexion --region oregon**

NOTE:

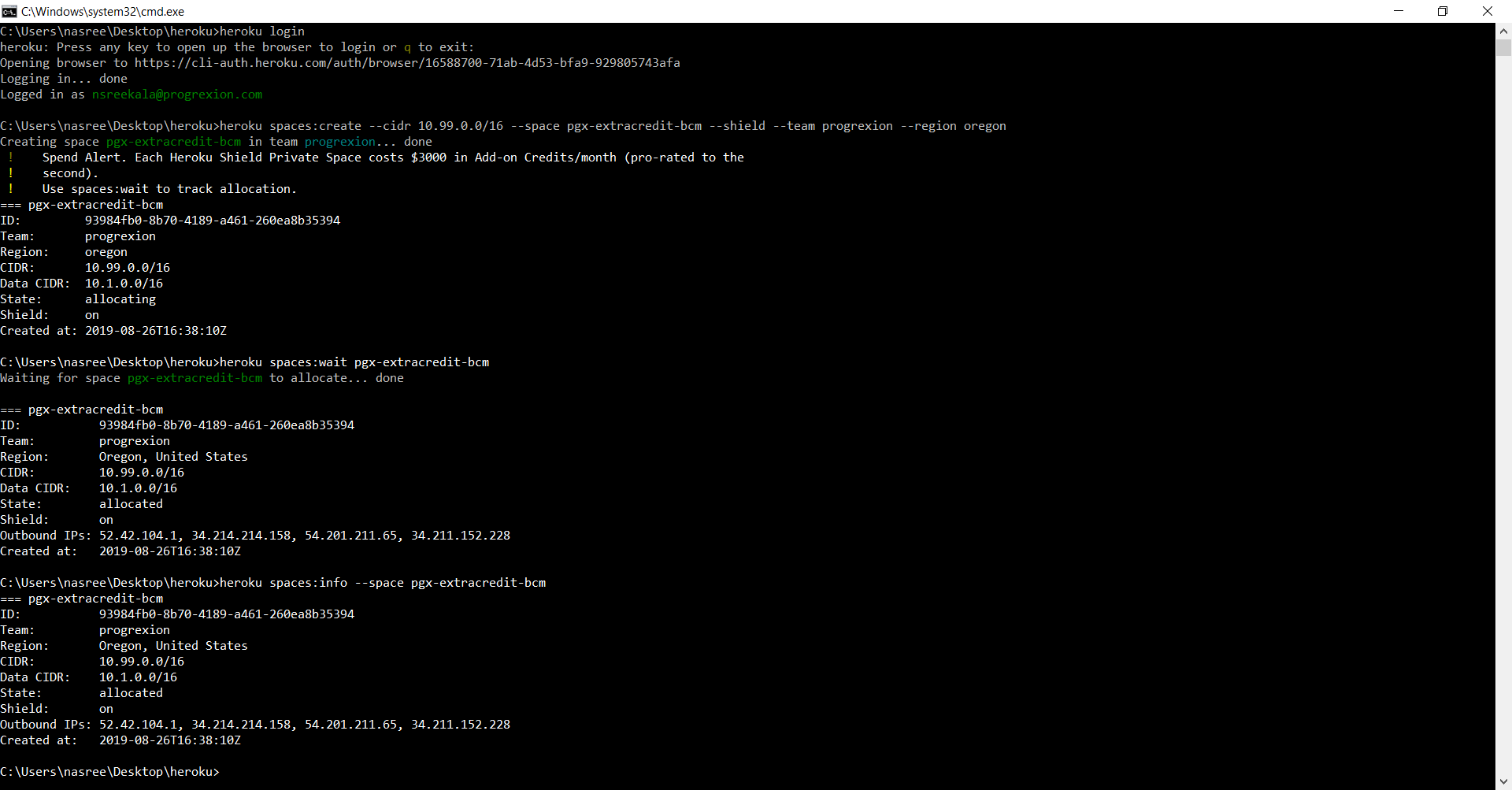
*CIDR*: 10.99.0.0/16 is the CIDR block range provided by Pgx

*Team*: ‘progrexion’ is the one used by Pgx

*Space Name*: ‘pgx-extracredit-bcm’ as chosen for BCM

*Region*: Provided ‘oregon’, mirroring the setting used for the other Private Space ‘pgx-production’ that is already in place under team ‘progrexion’.

Screenshot of the result is given below along with the “spaces:info” command output:



# Environment: Test

## Database

For BCM, we would be using the Amazon Aurora DB with PostgreSQL compatible version

#### Environment Details

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| db cluster name | Endpoints | DB name | Engine version | Multi AZ deployment | VPC |
| dbclusteridentifier-extracresit-bcm-test-data01 | [Writer]  dbclusteridentifier-extracredit-bcm-test-data01.cluster-czqcrnu8rq1a.us-west-2.rds.amazonaws.com  [Reader] dbclusteridentifier-extracredit-bcm-test-data01.cluster-ro-czqcrnu8rq1a.us-west-2.rds.amazonaws.com | BuildCredit | 10.7 | No | extracredit-dev-vpc-vpc (vpc-0ccd8c86dea26ace2) |

#### Configuration Steps

Configuration steps will be same as in Dev environment. Here the Cloudformation template name is ExtraCredit BCM Dev Deployment.

GitHub Location where the Cloudformation Templates for BCM are stored:

URL: <https://github.com/progrexion/extracredit-bcm-services.git>

Path: **./docs/cloudformation-templates**

## ActiveMQ

The same message broker is used for dev and test, hence the test MQ details will be same as in Dev environment.

## Heroku Shield Private Space

#### Environment Details

Same as in Dev environment.

#### Configuration Steps

Same as in Dev environment.

# Environment: Stage

## Database

#### Environment Details

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DB Cluster Name | Endpoints | DB Name | Engine version | Multi AZ deployment | VPC |
| dbclusteridentifier-extracresit-bcm-stage-data01 | [Writer]  dbclusteridentifier-extracredit-bcm-stage-data01.cluster-czqcrnu8rq1a.us-west-2.rds.amazonaws.com  [Reader] dbclusteridentifier-extracredit-bcm-stage-data01.cluster-ro-czqcrnu8rq1a.us-west-2.rds.amazonaws.com | BuildCredit | 10.7 | Yes | extracredit-stage-vpc-vpc (vpc-08de1c25ecd0ac599) |

#### Configuration Steps

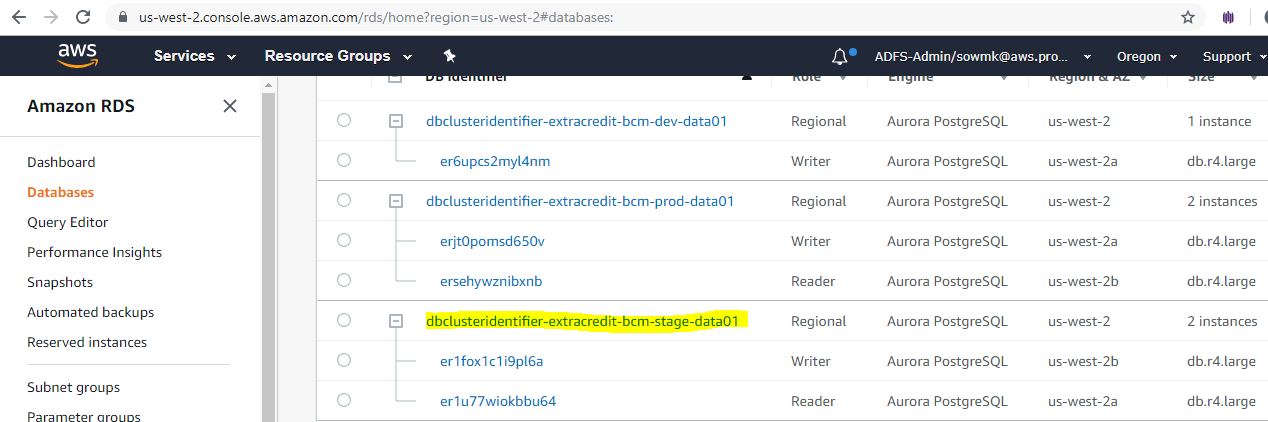
The Cloudformation template used here is ExtraCredit BCM Stage Deployment.

GitHub Location where the Cloudformation Templates for BCM are stored:

URL: <https://github.com/progrexion/extracredit-bcm-services.git>

Path: **./docs/cloudformation-templates**

Configuration steps would remain same as in Dev, except that in Stage, we use Multi AZ deployment with one read replica as shown in the below screenshot.



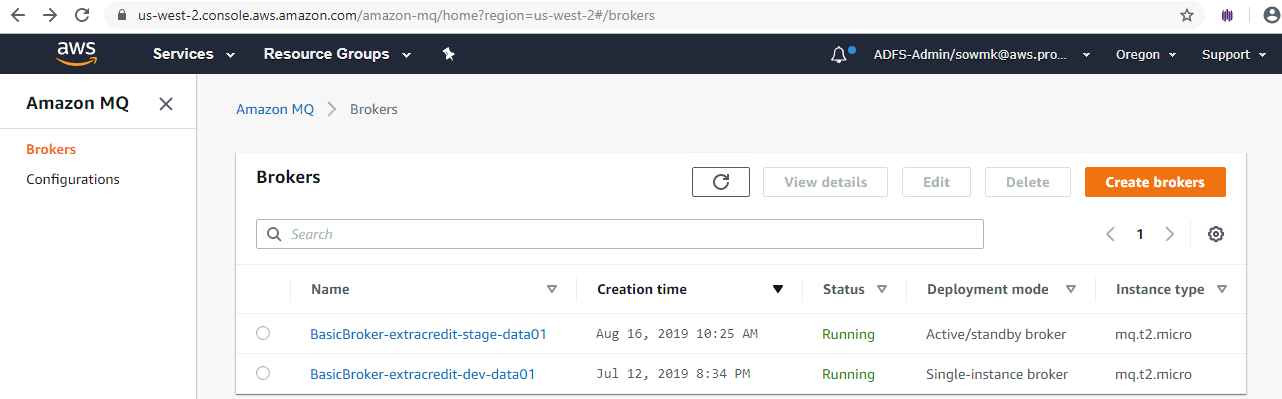
## ActiveMQ

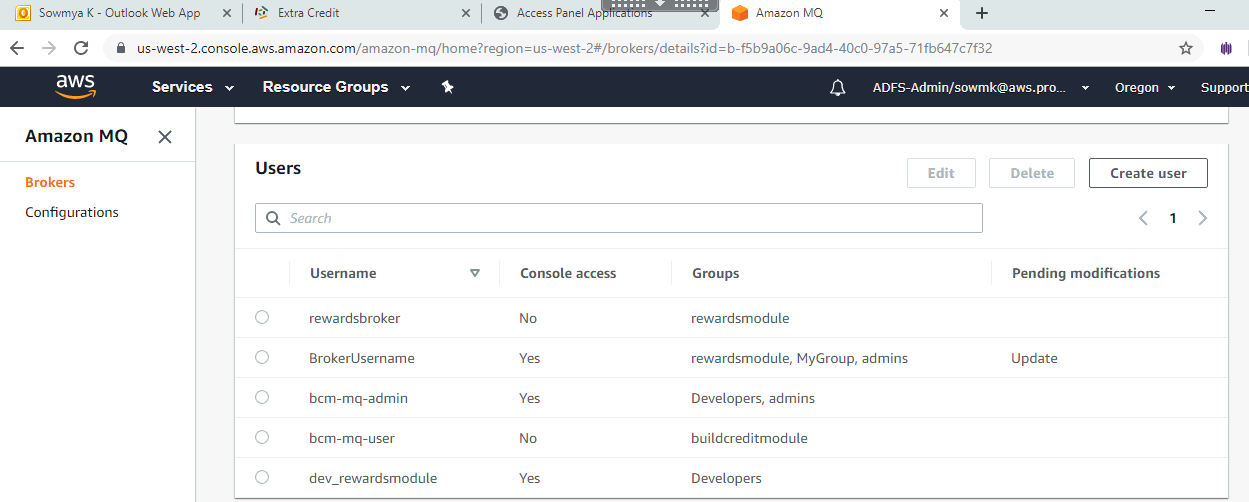
#### Environment Details

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Broker name | Broker instance type | Deployment mode | Active MQ Web console | Endpoint | Users created | Groups created |
| BasicBroker-extracredit-stage-data01 | mq.t2.micro | Active/Standby broker | <https://b-84e52d86-83dd-45da-b16e-b16d3ebaf755-1.mq.us-west-2.amazonaws.com:8162>  <https://b-84e52d86-83dd-45da-b16e-b16d3ebaf755-2.mq.us-west-2.amazonaws.com:8162> | ssl://b-84e52d86-83dd-45da-b16d3ebaf755-1.mq.us-west-2.amazonaws.com:61617  ssl://b-84e52d86-83dd-45da-b16d3ebaf755-2.mq.us-west-2.amazonaws.com:61617 | bcm-mq-admin  bcm-mq-user | Developers,admins,buildcreditmodule |

#### Configuration Steps

Amazon MQ for stage environment uses Active/standby broker for high availability. Normally, only one of the broker instances is active at any time, while the other broker instance is on standby. If one of the broker instances malfunctions or undergoes maintenance, it takes Amazon MQ a short while to take the inactive instance out of service, allowing the healthy standby instance to become active and to begin accepting incoming communications. When you reboot a broker, the failover takes only a few seconds.

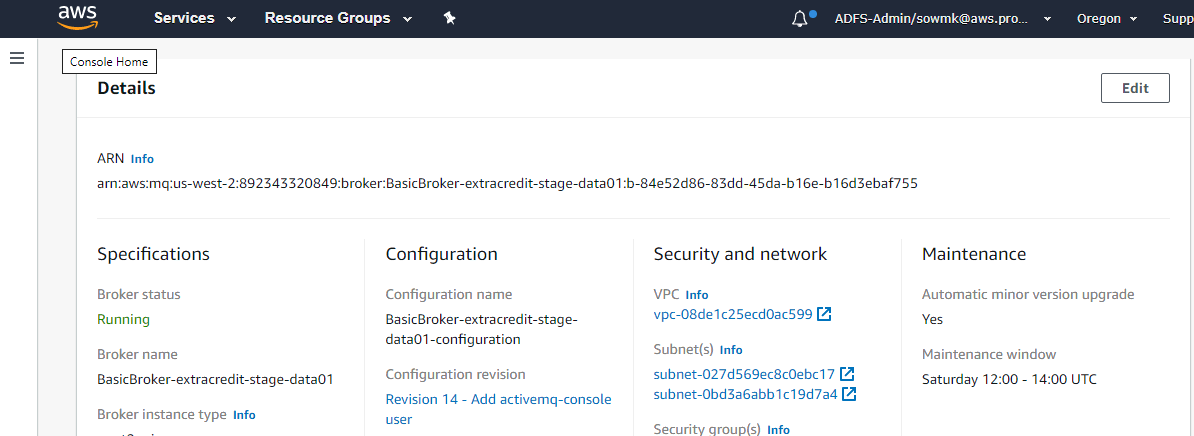




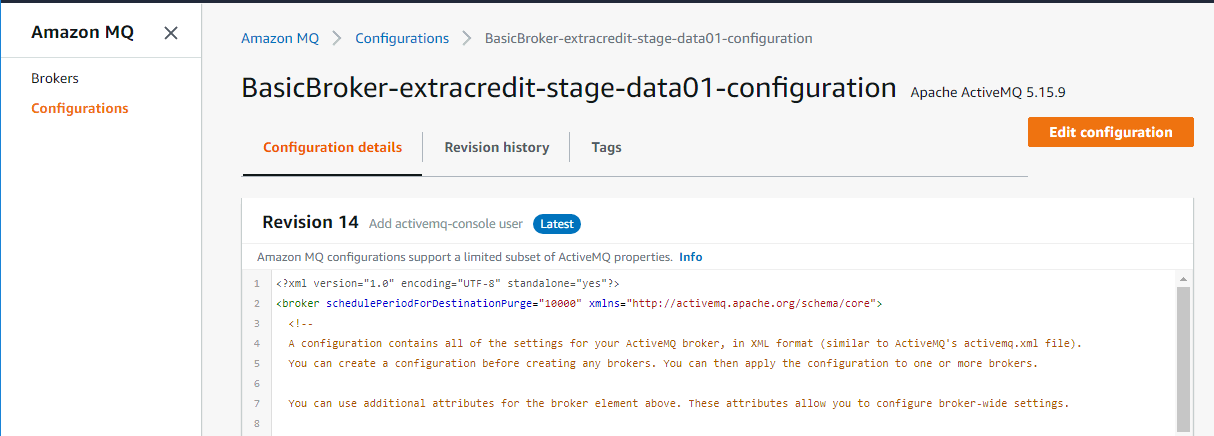
The Authorization plugin info has been modified to add the groups related to bcm module.



To change the Authorization plugin info navigate to Amazon MQ in the AWS console, select the relevant broker for stage (BasicBroker-extracredit-stage-data01) and in the details page under configuration revision click on the revision link



Click on edit configuration as shown in the below screen and make the necessary changes and save the changes.



Once the changes are saved, Edit the Broker, choose the Configuration Revision to apply, and Schedule the modifications. Opt for rebooting the broker immediately to get the changes take effect without waiting for the maintenance window.

## Heroku Shield Private Space

#### Environment Details

Same as dev

#### Configuration Steps

Same as dev

# Environment: Production

## Database

For BCM, we would be using the Amazon Aurora DB with PostgreSQL compatible version

#### Environment Details

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DB Cluster Name | Endpoints | DB Name | Engine version | Multi AZ deployment | VPC |
| dbclusteridentifier-extracresit-bcm-prod-data01 | [Writer]  dbclusteridentifier-extracredit-bcm-prod-data01.cluster-czqcrnu8rq1a.us-west-2.rds.amazonaws.com  [Reader] dbclusteridentifier-extracredit-bcm-prod-data01.cluster-ro-czqcrnu8rq1a.us-west-2.rds.amazonaws.com | BuildCredit | 10.7 | Yes | extracredit-prod-vpc-vpc (vpc-0cfeb673742bde6f2) |

#### Configuration Steps

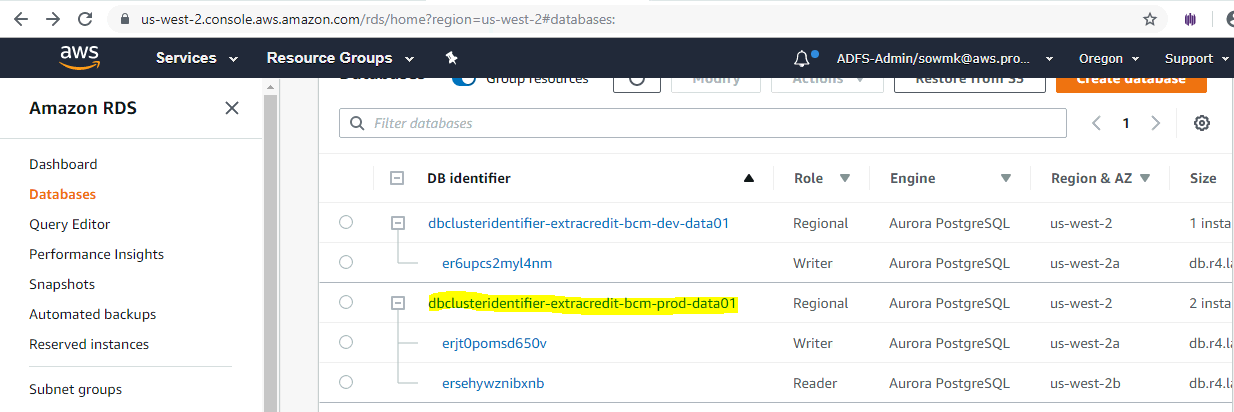
The Cloudformation template used here is ExtraCredit BCM Prod Deployment.

GitHub Location where the Cloudformation Templates for BCM are stored:

URL: <https://github.com/progrexion/extracredit-bcm-services.git>

Path: **./docs/cloudformation-templates**

Configuration steps would remain same as Dev, except that in Prod, we use Multi AZ deployment with one read replica as shown in the below screenshot.



## ActiveMQ

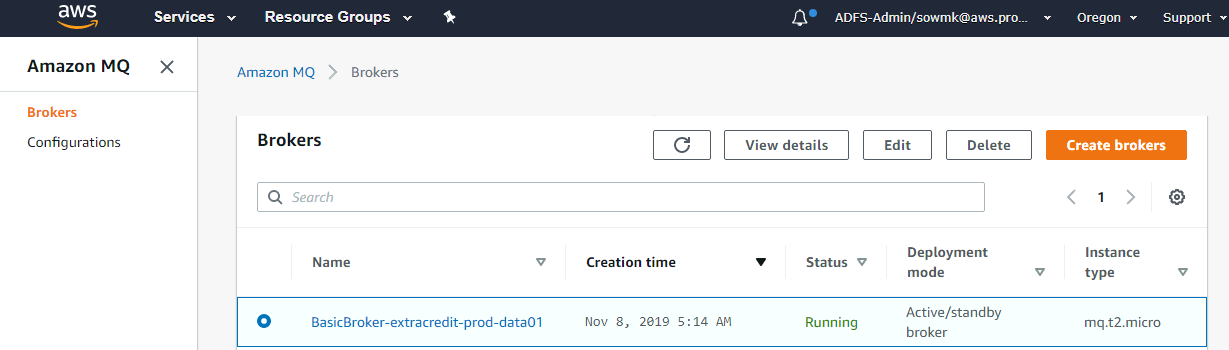
#### Environment Details

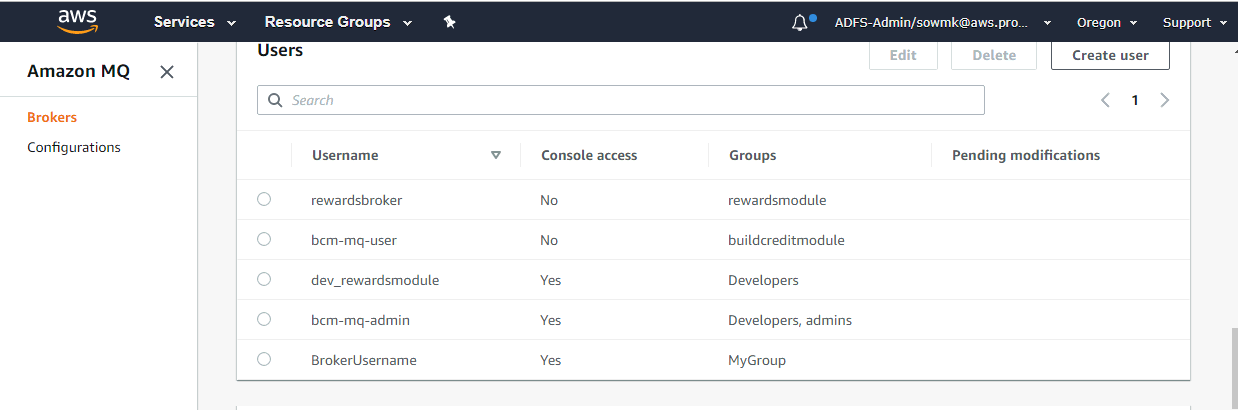
An Amazon MQ Broker was already made available in Prod as part of Rewards module setup, and BCM also uses the same broker to host the relevant Queue(s).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Broker name | Broker instance type | Deployment mode | Active MQ Web console | Endpoint | Users created | Groups created |
| BasicBroker-extracredit-prod-data01 | mq.t2.micro | Active/Standby broker | <https://b-d4262990-9c2e-4335-8271-f3173c12a267-1.mq.us-west-2.amazonaws.com:8162>  <https://b-d4262990-9c2e-4335-8271-f3173c12a267-2.mq.us-west-2.amazonaws.com:8162> | ssl://b-d4262990-9c2e-4335-8271-f3173c12a267-1.mq.us-west-2.amazonaws.com:61617  ssl://b-d4262990-9c2e-4335-8271-f3173c12a267-2.mq.us-west-2.amazonaws.com:61617 | bcm-mq-admin  bcm-mq-user | Developers,admins,buildcreditmodule |

#### Configuration Steps

Amazon MQ for prod environment uses Active/standby broker for high availability, similar to the one in Stage. Normally, only one of the broker instances is active at any time, while the other broker instance is on standby. If one of the broker instances malfunctions or undergoes maintenance, it takes Amazon MQ a short while to take the inactive instance out of service, allowing the healthy standby instance to become active and to begin accepting incoming communications. When you reboot a broker, the failover takes only a few seconds.

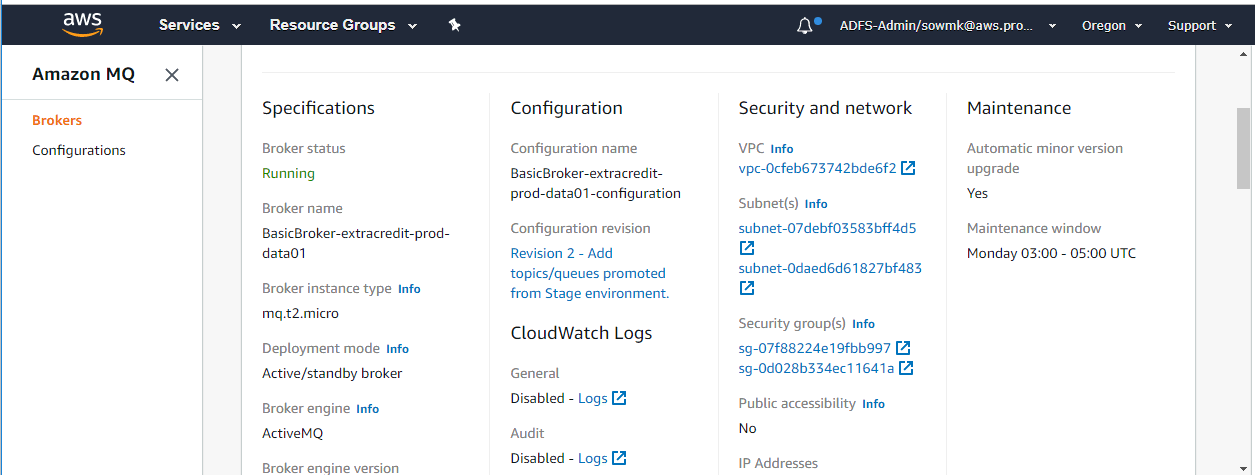




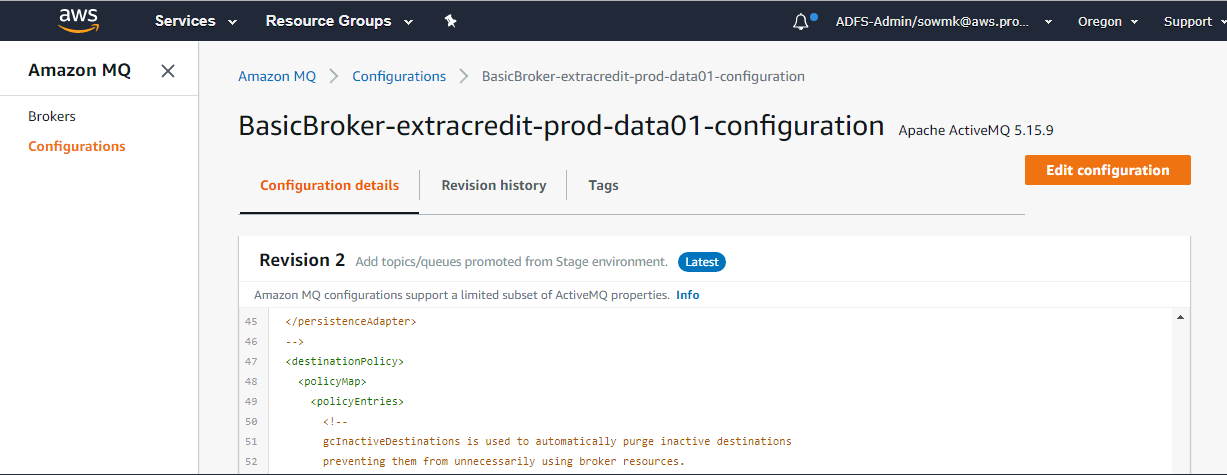
The Authorization plugin info has been modified to add the groups related to bcm module.



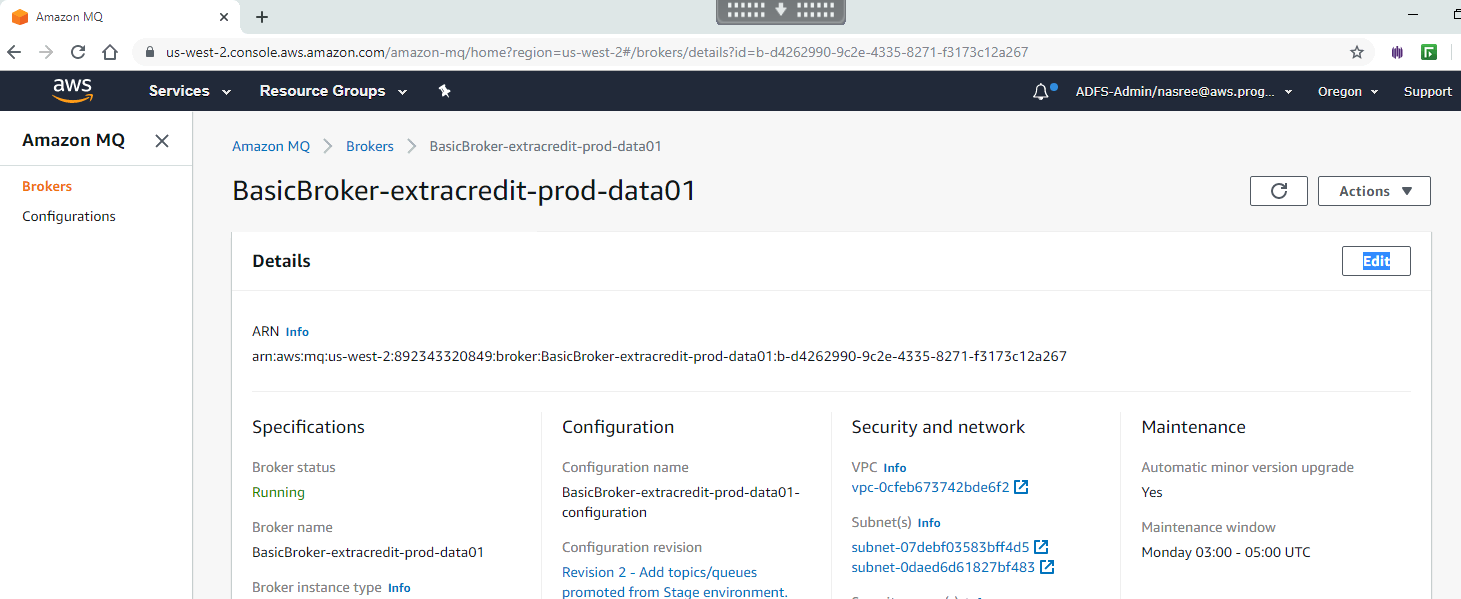
To change the Authorization plugin info navigate to Amazon MQ in the AWS console, select the relevant broker for prod (BasicBroker-extracredit-prod-data01) and in the details page under configuration revision click on the revision link.



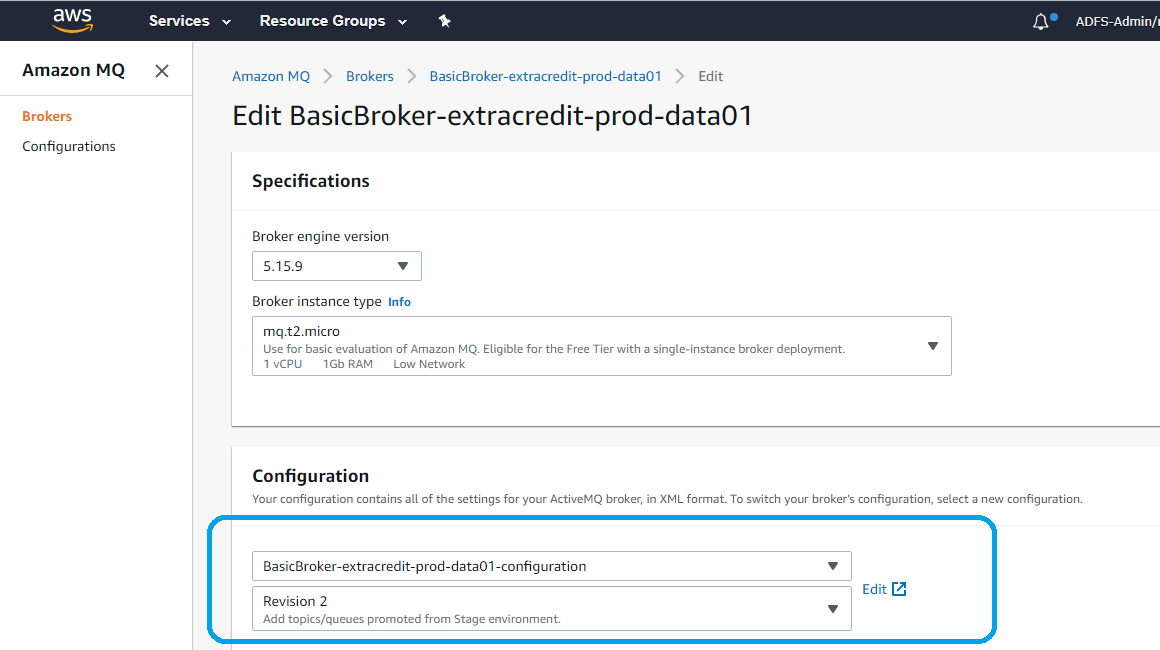
Click on edit configuration as shown in the below screen and make the necessary changes and save the changes.



Once the changes are saved, go back to the Broker details page > click Edit



Select the Configuration Revision from the drop down:



Click Schedule Modification. Opting for applying the changes by rebooting the broker immediately, instead of applying it on the weekly maintenance window, would help the changes to take effect immediately.

## Heroku Shield Private Space

#### Environment Details

Same as dev

#### Configuration Steps

Same as dev