DISASTER MANAGEMENT & RECOVERY

This document states how to take precautions if something fails in deployment. In case of backend and frontend deployment we always do tag based deployment, so if something fails or something went wrong in that we always revert to our previous deployed tag. But what if something happens with the database, how we can recover that, this document includes all this

NO.OF DATABASE = 3

1. Enveu_saas
2. Enveu_kill_bill
3. Mongo

HOW TO BACKUP AND RESTORE RDS

By default our RDS database make its automated backup in every 24 hours and we have to take backup after every 4 hour, so in case of RDS we are having two databases i.e **enveu-saas** and **enveu-kill-bill**, so we will write a script for the backup process and we will give the path of that script in crontabs and we will provide the cron expression in our crontab along with our script path.

The script that we have written is named as **rds-snapshot.sh** and is as follows:-
To make it executable give it permissions **chmod +x rds-snapshot.sh**

```
#!/bin/bash

aws rds create-db-cluster-snapshot \
    --db-cluster-identifier enveu-saas \
    --db-cluster-snapshot-identifier enveu-saas-snapshot$(date "+%H-%M-%S")


aws rds create-db-cluster-snapshot \
    --db-cluster-identifier enveu-kill-bill \
    --db-cluster-snapshot-identifier enveu-kill-bill-snapshot$(date
"+%H-%M-%S")
```

As you can see we have provides date argument at the end so that we should know at what time the backup was done.

To put this in crontab execute:-

```
crontab -e
0 */4 * * * /home/ubuntu/rds-snapshot.sh
```

Here we have entered the cron expression that will execute our script after every 4 hour, when it will execute you will get a response like this:-

```
{
    "DBClusterSnapshot": {
        "StorageEncrypted": false,
        "DBClusterSnapshotArn":
"arn:aws:rds:us-east-1:687679337356:cluster-snapshot:enveu-saas-snapshot06-
20-34",
        "AllocatedStorage": 1,
        "ClusterCreateTime": "2019-04-05T20:26:31.009Z",
        "DBClusterSnapshotIdentifier": "enveu-saas-snapshot06-20-34",
        "LicenseModel": "aurora",
        "SnapshotType": "manual",
        "VpcId": "vpc-029e3c78",
        "MasterUsername": "enveu_saas",
        "PercentProgress": 0,
        "DBClusterIdentifier": "enveu-saas",
        "AvailabilityZones": [
            "us-east-1b",
            "us-east-1c",
            "us-east-1e"
        ],
        "Port": 0,
        "EngineVersion": "5.6.10a",
        "SnapshotCreateTime": "2019-10-11T06:20:35.372Z",
        "Status": "creating",
        "Engine": "aurora",
        "IAMDatabaseAuthenticationEnabled": false
    }
}
{
    "DBClusterSnapshot": {
        "SnapshotType": "manual",
        "LicenseModel": "aurora",
        "AllocatedStorage": 1,
        "DBClusterSnapshotArn":
```

```
"arn:aws:rds:us-east-1:687679337356:cluster-snapshot:enveu-kill-bill-snapsh
ot06-20-35",
        "Port": 0,
        "EngineVersion": "5.6.10a",
        "PercentProgress": 0,
        "KmsKeyId":
"arn:aws:kms:us-east-1:687679337356:key/a27ddd97-6d22-4439-a5c5-188b37f6089
b",
        "MasterUsername": "root",
        "Engine": "aurora",
        "Status": "creating",
        "DBClusterSnapshotIdentifier": "enveu-kill-bill-snapshot06-20-35",
        "DBClusterIdentifier": "enveu-kill-bill",
        "ClusterCreateTime": "2019-08-19T12:46:51.263Z",
        "AvailabilityZones": [
            "us-east-1c",
            "us-east-1d",
            "us-east-1e"
        ],
        "IAMDatabaseAuthenticationEnabled": false,
        "StorageEncrypted": true,
        "VpcId": "vpc-029e3c78",
        "SnapshotCreateTime": "2019-10-11T06:20:35.955Z"
    }
}
```

This means that your backup has been created and always remember that while performing a backup task your database should always be in available state

You can find the created backup in the snapshot column of your rds along with the time it was created, from there it can be restored

To know how to restore your database with the created snapshot, follow the link below:-
https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_RestoreFromSnapshot.html

That's about how we can backup and restore our rds database

HOW TO BACKUP AND RESTORE MONGODB

We are using mongo db using docker and docker-compose, the docker-compose for mongo is as follows:-

```
version: '3.1'
services:
  mongo:
    image: mongo
    restart: always
    volumes:
          - /var/lib/data/db:/data/db
    ports:
      - 27017:27017
    environment:
      MONGO_INITDB_ROOT_USERNAME: <username>
      MONGO_INITDB_ROOT_PASSWORD: <password>
```

We have mounted a volume to our docker container as you can see in the docker-compose which means that all the data that mongo will generate will be sent to that location that is in docker-compose i.e. **/var/lib/data/db**

We want to take the backup of our mongodb after every 4 hours, so we have written a script that will create a folder with the particular date and time in it and the backup of our mongo will go in that folder or directory and it will make tar of that backup and from there we will move our backup tar file to s3 with the same script because if we will have the backup on our server than our server disk will get full, so we will move our backup to s3 bucket, the script that we have written is as follows:-

```
#!/bin/sh

cd mongo-db-backup

mkdir $(date +%Y_%m_%d)

cd $(date +%Y_%m_%d)

mkdir $(date +%H:%M)

cd /var/lib/data/
```

```
sudo cp -r db/ /home/ubuntu/mongo-db-backup/$(date +%Y_%m_%d)/$(date
+%H:%M)

cd /home/ubuntu

cd mongo-db-backup/$(date +%Y_%m_%d)/$(date +%H:%M)

sudo chown -R ubuntu:ubuntu db/

tar -zcf db-backup-tar.gz db/

sudo rm -r db/

cd /home/ubuntu

#### MOVING MONGO BACKUP TO S3 BUCKET ######

sudo aws s3 cp mongo-db-backup/ s3://enveu-db-backups/mongo-db --recursive

cd /home/ubuntu/mongo-db-backup

sudo rm -r *
```

To make it executable we need to give it permissions for **chmod +x**

After that we need to put this script into our crontab as we did earlier in case of RDS

When all these things will be done, crontab will execute our script and it will take the backup of our mongo and it will make tar of that backup and will move that to s3.

In case we want to restore our backup we need to wget that particular tar file from s3 and we need to untar it and after that we have down our docker-compose(mongo) and we need to copy the untar data an copy that to our volume location, but first you have to remove the old data from there, after that you have to do **sudo docker-compose up** and your mongo will be up with the restored data.