**IOT CA3**

**The Smart Coaster (Beta Release)**

A logo with coffee beans on it

Description automatically generated

Jamie Lawlor

Caitlin Maguire

Luke Hilliard

Shahzad Shabeer

**Version 1**

**December 2024**

**Table Of Contents**

1. [**SYSTEM ARCHITECTURE……...…………………………...…………………………... 3**](#SystemArchitecture)
2. [**DEPLOYED TO AWS…….………………………………………………….......................**](#AWSAlpha)

**2.1** [Custom Domain..…………………………………………………………………………..……….](#CustomDomain)

**2.2** [HTTPS……….…………………………………………………………………………..................](#AWS2)

**2.3** Available after submission………………………………………………………………………….

1. [**PUBNUB……………………...………………………….…………………………….…….**](#Pubnub)
   1. [Access Manager…………………………………………………………………………...….……](#Communication)
   2. SSL…………………………………………………………………………………………………
   3. Communication…………………………………………………………………………………….
2. [**SECURITY……………………...……………………….……………………………….….**](#Security)
   1. [Device………..……..……………………………………………………………………...………](#IoTDevice)
   2. [Communication Channels……………………...…………………………………………………..](#CommunicationChannels)
   3. [Database…………………………………………………………………………………...……….](#DatabaseSecurity)
   4. [Webserver…………………………………………………………………………………………](#ServerSecurity)
   5. [Data in flow……………………………………………………………………………………..](#DataInTransit)
3. **System Architecture**

**2. Deployed to AWS**

* 1. **Custom Domain – Jamie**

**2.2 HTTPS – Jamie**

**2.3 Available after Domain – Jamie**

**3. Pubnub**

**3.1 Access Manager – Jamie**

**3.2 SSL/TLS – Jamie**

**3.3 Communication – Jamie**

**4. Security**

**4.1 Device –**

* 1. **Communication channels – Jamie**
  2. **Database – Caitlin**

A large part of database security is having a backup running of the database. This can be easy to do using a MySQL database, however we are using a NoSQL database therefore this was more difficult. The cluster I created for the project is under a free tier, within the free tier you cannot access the built in backup features of MongoDB Atlas.

Following some research I have read up about different methods to back up a mongo database, and I decided to follow the approach of this website:  <https://medium.com/@andrewskangah/how-to-create-a-middle-man-automatic-backup-for-mongodb-atlas-mo-sandbox-cluster-tier-5e7276d66ee2>

First of all, I created a local backup of the database on my desktop. I did this by creating a folder called mongobackup, in the terminal I went into this location and pasted in the export connection string from the MongoDB Atlas website. In the connection string I included my username and password for the admin user, as I have full access to the database. I also needed to edit the connection string by including the database name and the specified collection, I started with the drink status collection.

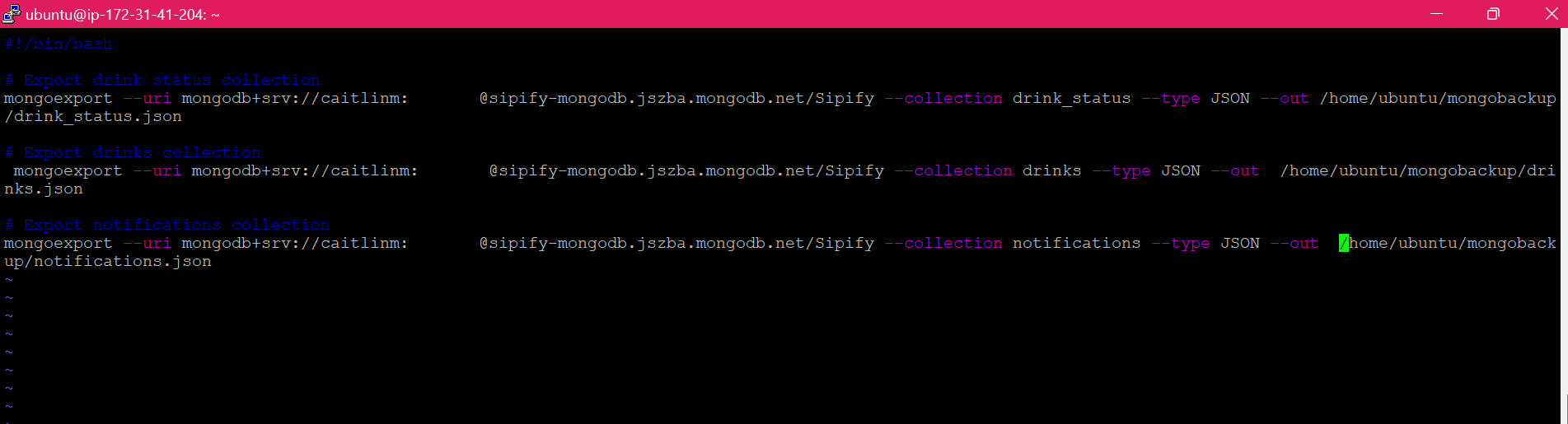
**Local Database Backup:**

**A screenshot of a computer

Description automatically generated**

After adding in the connection string in the terminal, I was getting an error that mongo-export was not recognised. I researched how to get mongo export installed in the terminal, and found I can download the MongoDB tools from the mongo website. I downloaded the tools into  my program files and created a path in the environment variables to the mongo tools. Once I did this the mongo export command was recognised and I was connected, the drink status records were coming into the local mongobackup folder. I repeated this process for my other collections drinks and notifications.

Once I was able to connect to atlas and create a local backup, I started the process of creating a backup on the AWS instance using a cron job. To do this first I created the script file called mongobackup.sh.

**Script:**

Next, I ran the script file to ensure it was connecting properly to atlas, then I moved on to creating the cron job. To test that the cron was running as it should, I set it to run every minute.

**Cron:**

**A screen shot of a computer

Description automatically generated**

**Log:**

To check the backup was running correctly I ran a command to view the syslogs.

**A screen shot of a computer

Description automatically generated**

 From the logs we can see the cronjob is running every minute as it should, now to check that the correct contents have come in to the mongobackup folder.

**Mongobackup Folder:**

**A screen shot of a computer

Description automatically generated**A copy of each of the files has been added the drink status , drinks and notifications all for each of the collections in the database.

**MongoDB Atlas Drink Status Collection:**

**A screenshot of a computer

Description automatically generated**

**Drink Status JSON:**

**A screen shot of a computer screen

Description automatically generated**

 By comparing the records from the drink status collection in MongoDB Atlas and in the backup drink\_status.json, we can see that the records have come in correctly . There are 17 records in the database and 17 values in the Json.

Now to test the backup, I am going to add in a new record into the MongoDB Atlas and check to see if it is added into the next backup on the server.

**New Record:**

**A screenshot of a computer

Description automatically generated**

A new record with the drink type Mocha has been added to the drink status collection, now we wait one minute to ensure the record is added to the next backup of the database in the server using the cron job.

A screen shot of a black screen

Description automatically generated

In the logs we can see the backup has ran again as expected.

**Updated Drink Status JSON:**

A screen shot of a computer

Description automatically generated

The bottom record shows that the new drink "Mocha" has been added into the back up of the database, using cron.

As we are not dealing with a large set of data, I think an appropriate number of backups would be once a week. I have changed the cron job to run every Friday at 12am as shown below.

**Updated Cron: A screenshot of a computer

Description automatically generated**

I used the website <https://crontab.cronhub.io/> to learn how to use cron, in the code above a Sudo command will be run on the script file that I created. The script file includes each of the mongo export connection strings to the collections. Without calling the sudo command on the script file, the cron job was not running correctly, and the records were not coming in.

For our database security we now have both a local backup and a backup running on a cron job on the AWS instance. By using the backup on the instance this ensures the drink status collection will be updated as new drinks are added. The drinks and notification collections are also included in the backup; however, the two collections will most likely not be changed unless necessary as this data is used as a comparison for the drink status.

**Drinks Collection Backup:**

**A screen shot of a computer

Description automatically generated**

**Notifications Collection Backup:**

**A screen shot of a computer

Description automatically generated**

 For the security of the database I have also created different users in Atlas, I am set to the admin user, and I have created users for my team members.

**Users In Atlas: A screenshot of a computer

Description automatically generated**

**4.4 Webserver – Jamie**

**4.5 Data in flow - Jamie**