COSC349 Assignment 2 Report  
  
Samuel Ng 2955262  
Matt Mouat 7552560

For this assignment, we opted to utilize our first one, building upon it to generate a database that can both save and retrieve data. How this application is intended to be used is that a user can upload data of various kinds, which we have currently kept to basic numbers and letters, and retrieve it via a separate website, arranged in an orderly manner. We utilized three virtual machines in our first assignment to demonstrate our application, but as the AWS (Amazon Web Service) service provides storage, we changed our number of machines to two. For our storage choice, we opted for the RDS (Relational Database Service) database, as it had support for MySQL with little change to the code. We have also utilised the security benefits of Amazons VPC (virtual private cloud) services to be able to limit the traffic to and between our VMS as to increase security and maximize the resources we rent.

Our virtual machines are deployed through Vagrant, using the AWS plugin. With some alterations to the Vagrant file, our application would be easily deployed through a simple git bash terminal, and thus we opted for this route. By providing the Vagrant file with our AWS credentials, our virtual machines would be uploaded as EC2 instances, and be automatically setup without much hassle. The database itself is set up manually through AWS, and thus the only subject of worry was the errors thrown by any scripts or lines of code.

For accessing our database query and retrieval, there will be two websites up, and thus two links. These are provided in the GitHub <https://github.com/matt53211/Cosc349asgn2>, and can be used to enter data or retrieve data. Provided in the GitHub are also instructions if you wish to set this up for yourself, either on your local computer or on a cloud server. There are in-depth instructions for both, including a disclaimer on the operating system an individual may be using.

Although there were multiple options for storage such as Aurora and MySQL, we persisted with the RDS database as they provided multiple types of SQL servers. RDS databases have an added benefit of deploying MySQL servers within a short period, in a cost-efficient manner, which are all priorities of a simple database. The data is stored in a simple table, which can then be retrieved and displayed via another website. The way the data is stored is relatively insecure, due to our style of handling.

Difficulties

There were multiple issues going forward from our development. The first issue we had to tackle was altering the Vagrant file. As the AWS plugin had specific requirements, we had to first figure these out before adding them into the Vagrant file, ensuring we had input the right credentials. The second issue we had to tackle was the development and alteration of the PHP files. These files, as they were based upon our first assignment, were considerably underdeveloped and thus needed work. On top of that, we were caught off guard by the need to specify a webpage within the PHP files themselves, which led to more issues. Finally, we had multiple errors, mostly to do with the credentials, and permissions in the shell scripts. Even after fixing credential errors, there were many self-wrought errors that lead to problems down the line, such as invalid syntaxes and wrong comments used. Even after rectifying these errors, we had some name mismatches that went unnoticed for a long time, as the website technically loaded. These were quickly rectified once noticed.

Listed below are our considerations and how we set up our files.

Database Possibilities

We use a RDS for our database as it is the cheaper option for basic data storage, and it comes with a wide range of useful tools for organisation and finding data.

RDS Pros and Cons

RDS is made for information that is unlikely to change much if at all, so our contact information recording program is well within the bounds. This also means that we can utilise the RDS’s ability to sort and index the information quickly to allow for massive scalability, but due to its strict regulations of data that is rarely changed, it makes for a poor database to choose when considering repurposing as it severely limits the changes that can be made.

Pros

* RDS can roll back an operation if something goes wrong whereas S3 needs a separate program to go back a step.
* RDS may be slightly more expensive than S3 for storage, but it comes with a wide range of sorting, searching and other useful tools which would have to be acquired separately if S3 is used instead. These wide range of tools makes it cheaper in the long run.

Cons

* RDS has limited flexibility when it comes to altering the program to needs other than text or numeral based recording software

Prices (USD)

S3

* 0.025$ per GB up to 50T.
* Retrieving and insertion costs are 0.005$ per 1000 requests.
* Free downloads up to 1GB a month, then 0.009$ per GB for next 10T.

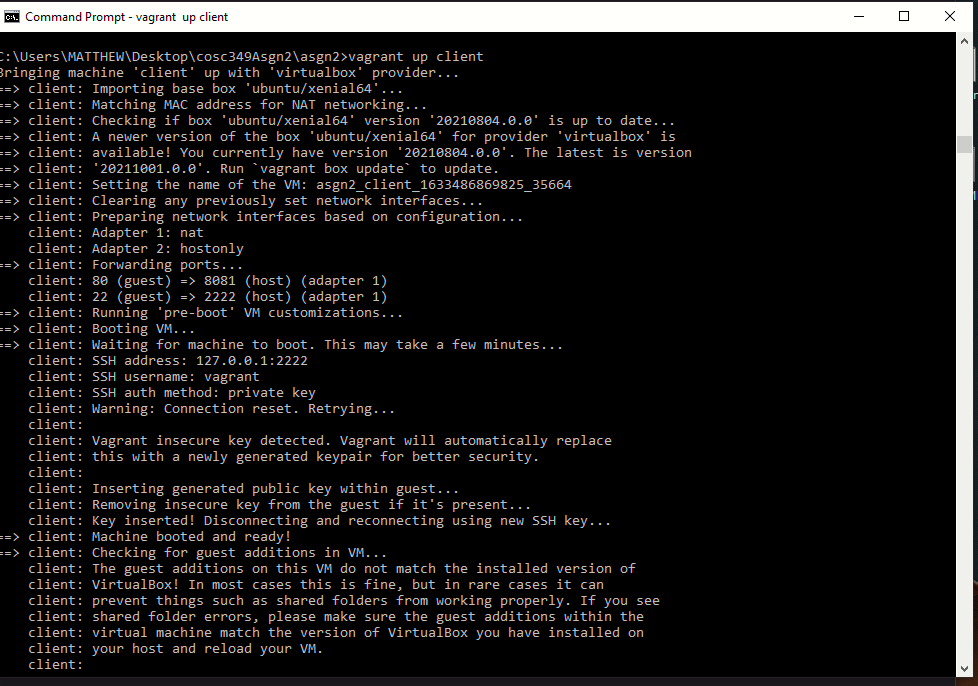
RDS

* Varies depending on engine, currently using Aurora’s prices.
* 0.1$ per GB of storage per month.
* 0.2$ per million requests at s3 rates that’s 5$ per million.
* Low storage usage regardless as its just set text data.

Cost estimation

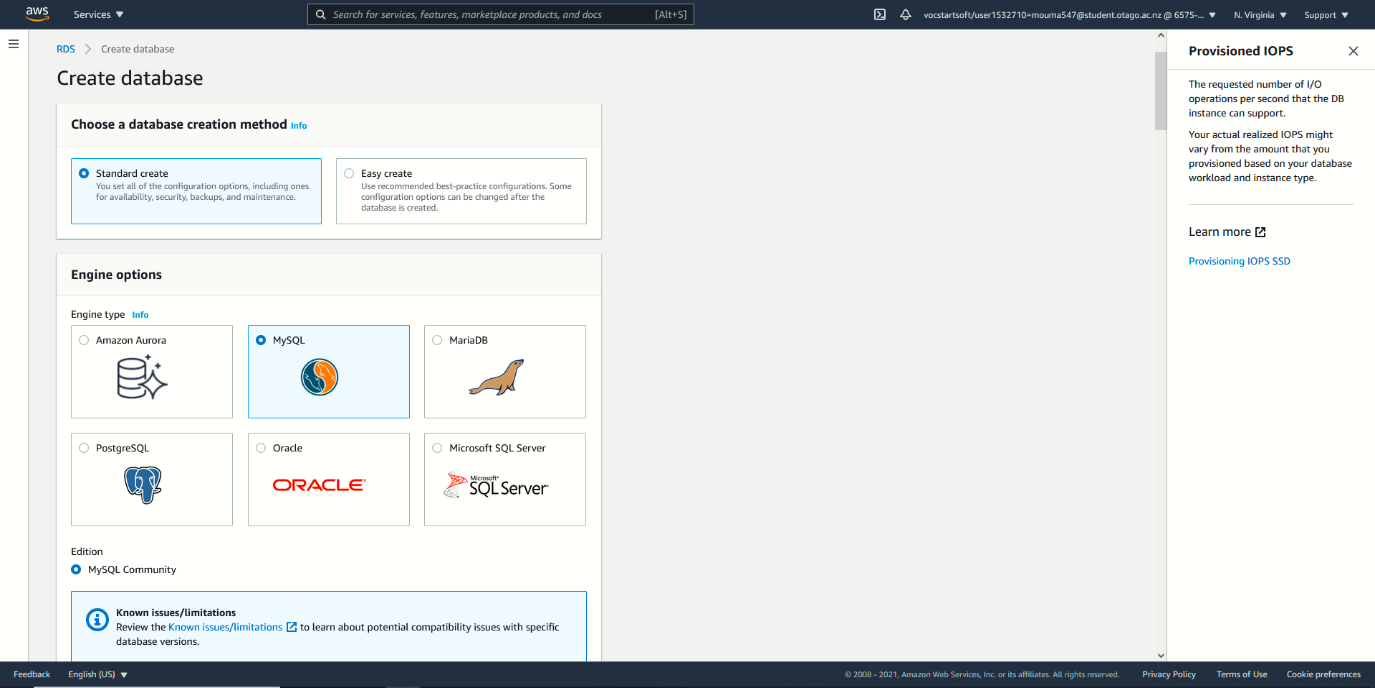
* Cost of 2 EC2 VM’s is $13.72 USD per month. We add this to the monthly cost of low usage RDS ($2.48 per month) and the low estimate running cost for idle is $16.20 USD per month.
* During light usage (40% or less) the cost estimation is around $30.57 USD a month.
* Cost estimation was calculated using Amazon’s calculator. https://calculator.aws/#/createCalculator/EC2

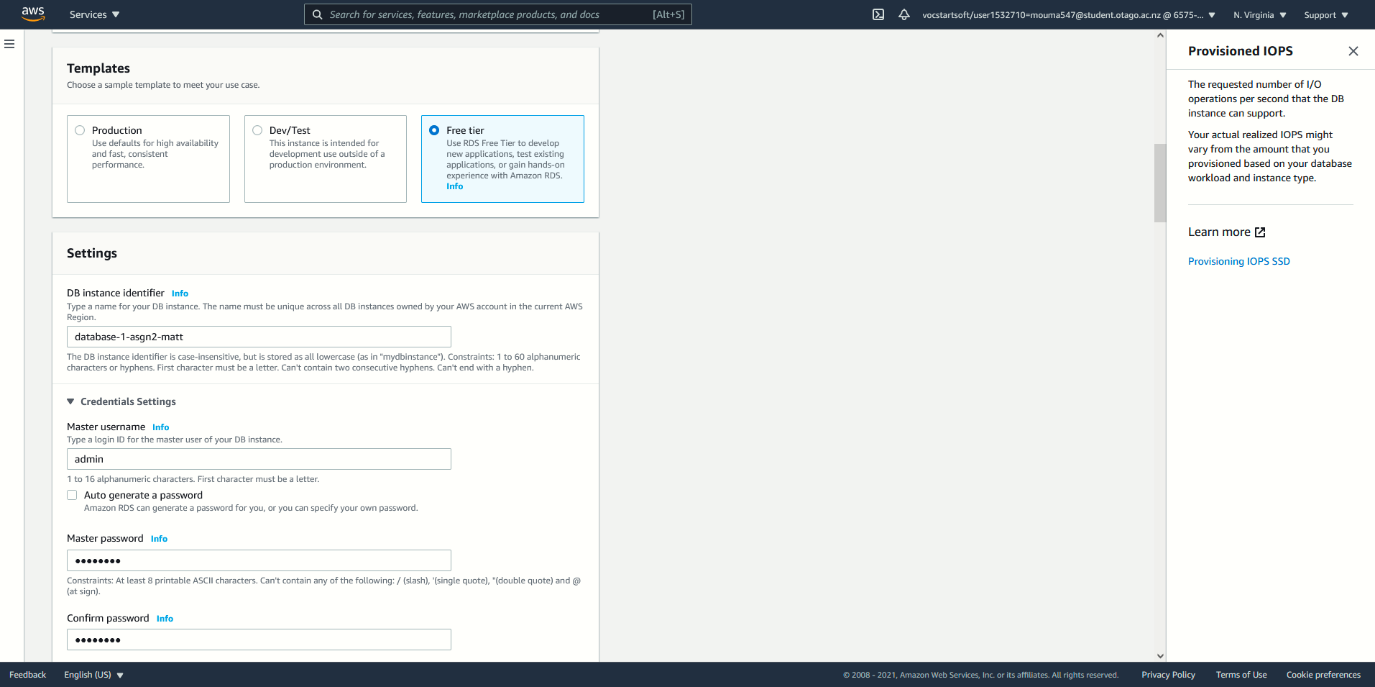
Setup

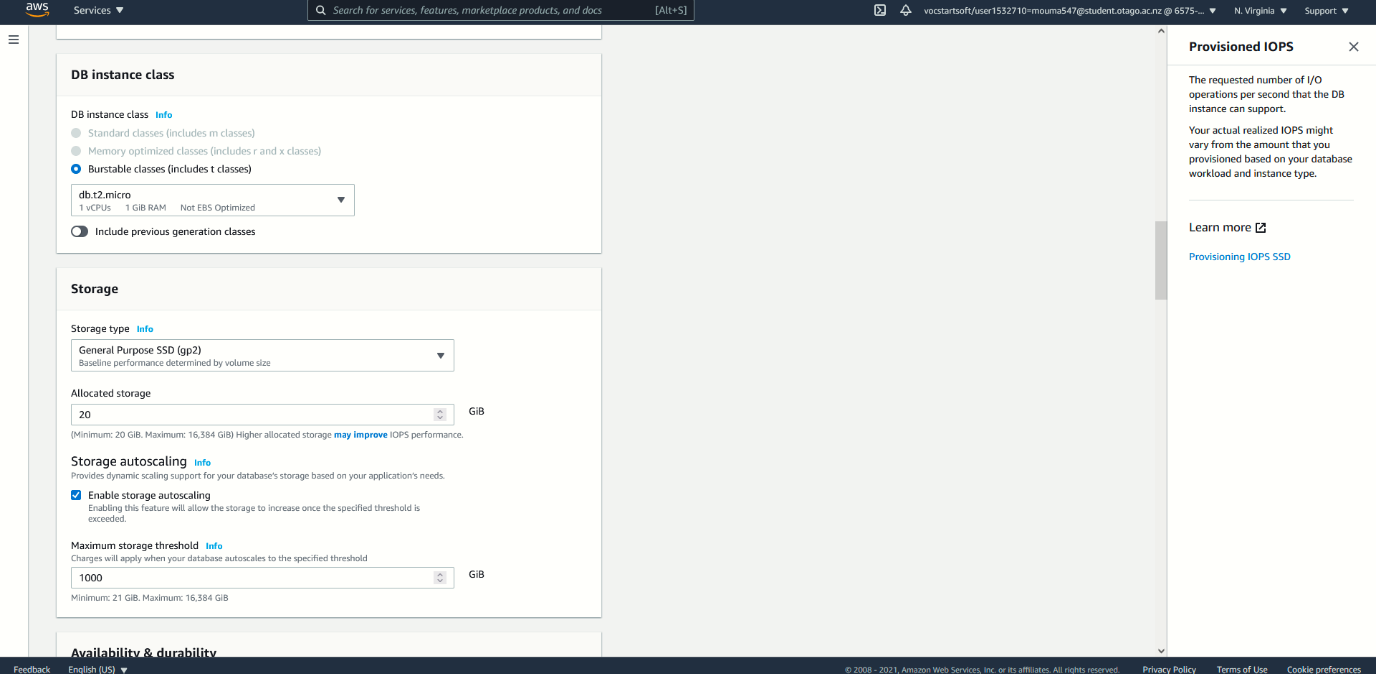
Running Vagrant up

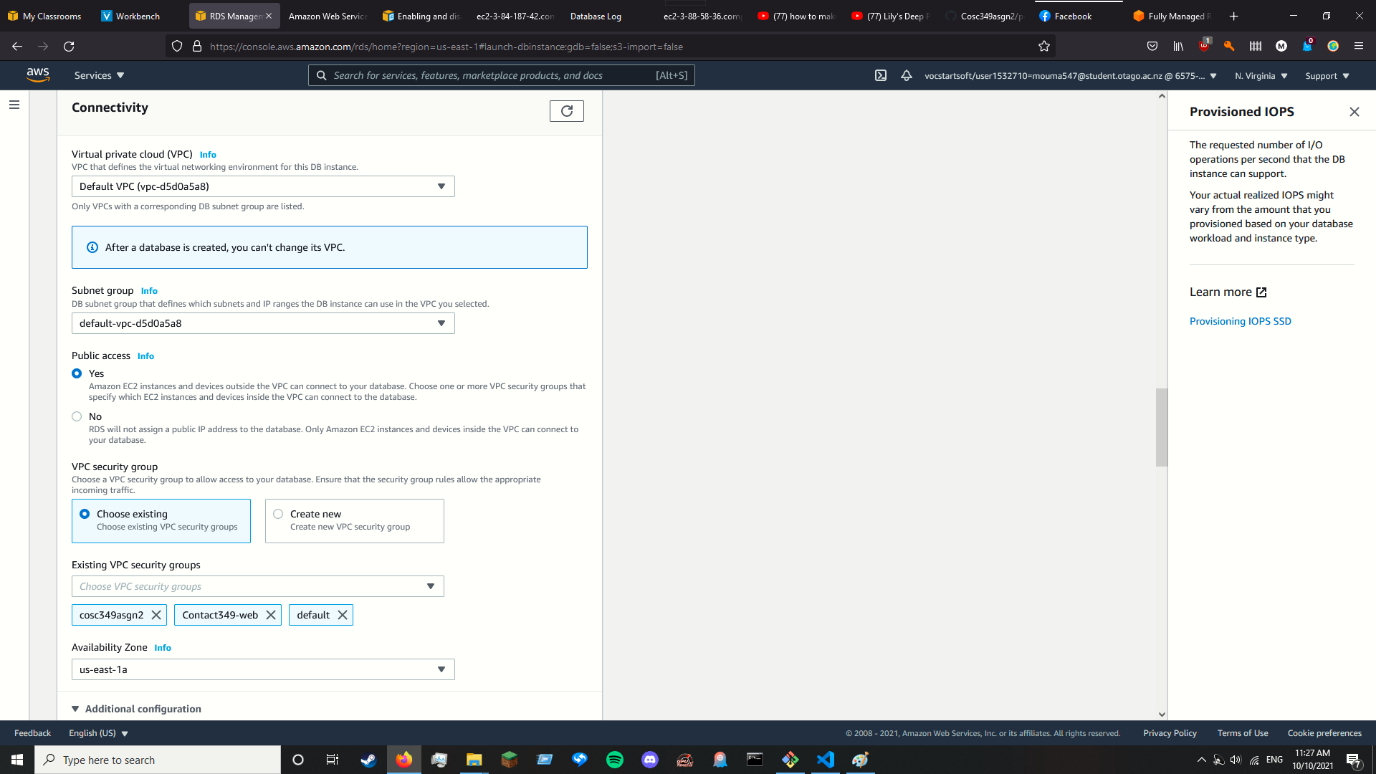
Setting up the RDS

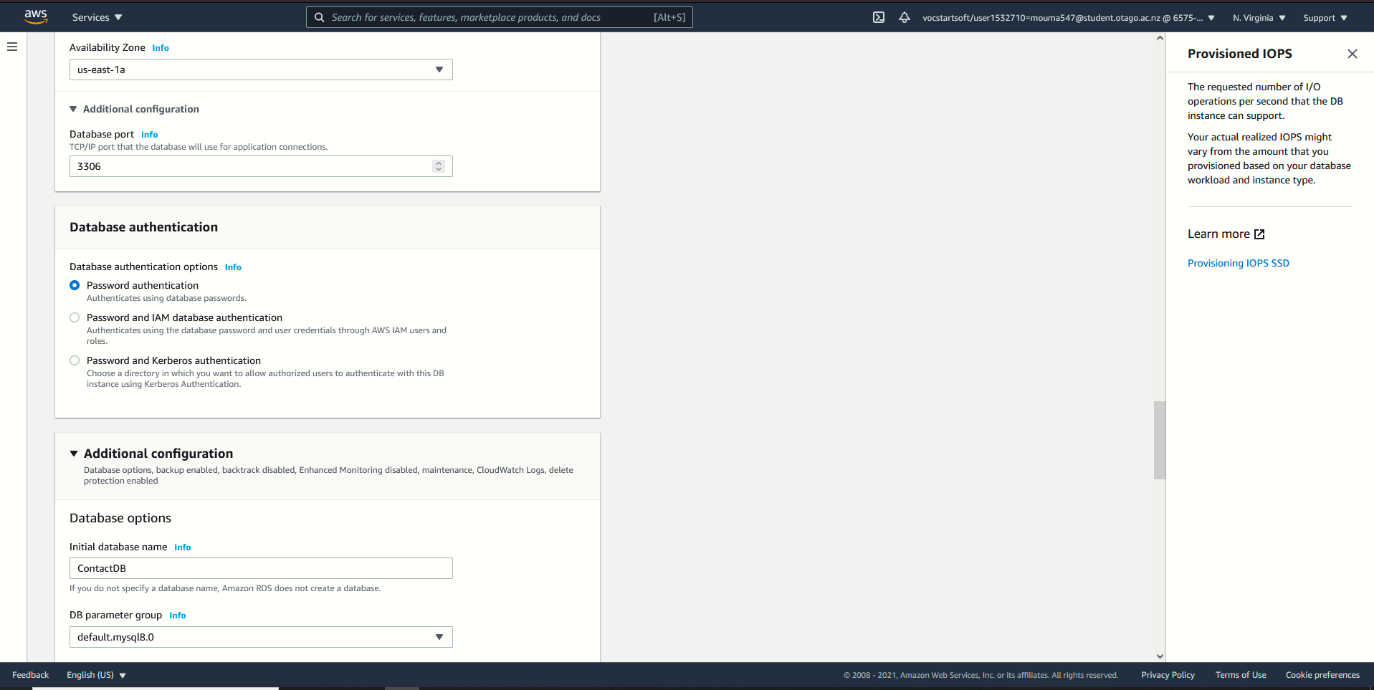
Using MySQL as it has the most compatibility with the original database setup.



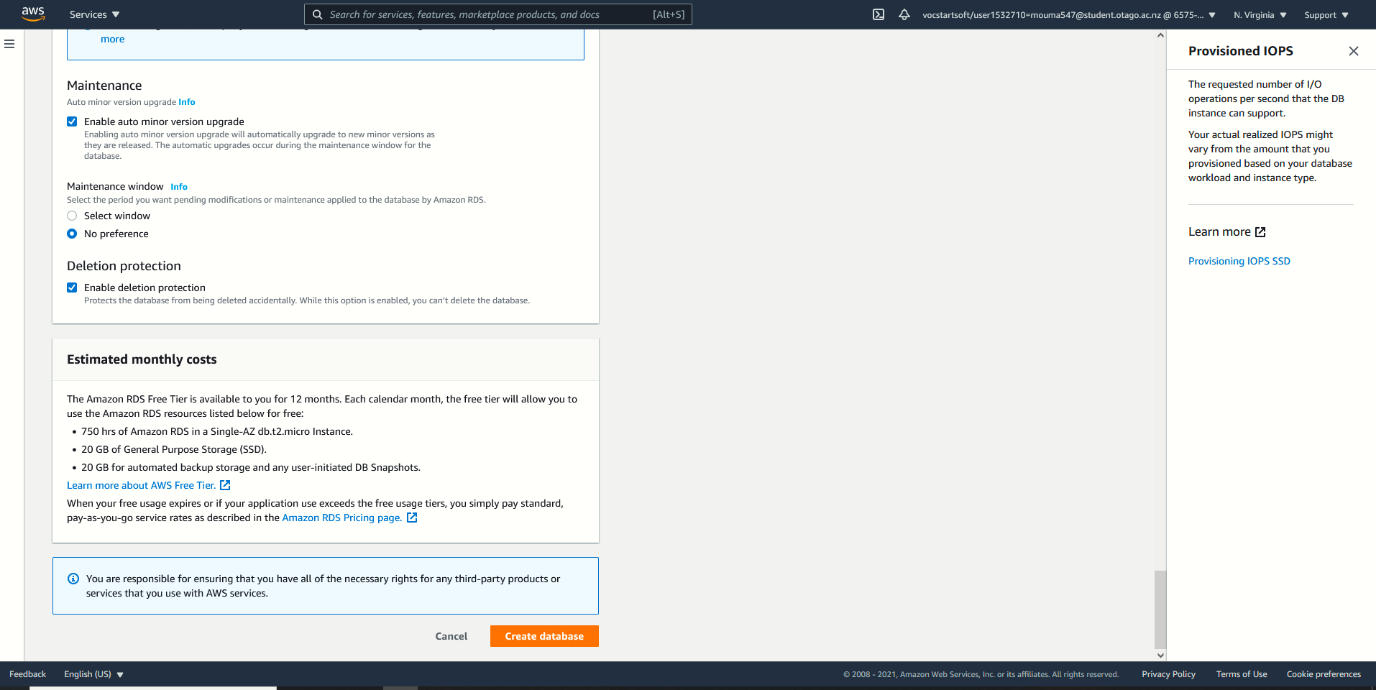
Choosing the free tier as we want to use as little money as possible for presenting our product online.

We are using a “db.t2.micro”. For our current usage expectations, it is the most efficient cost to capacity ratio that we can attain. Bigger companies will need to go for amazon’s larger scale servers which unfortunately cost more.

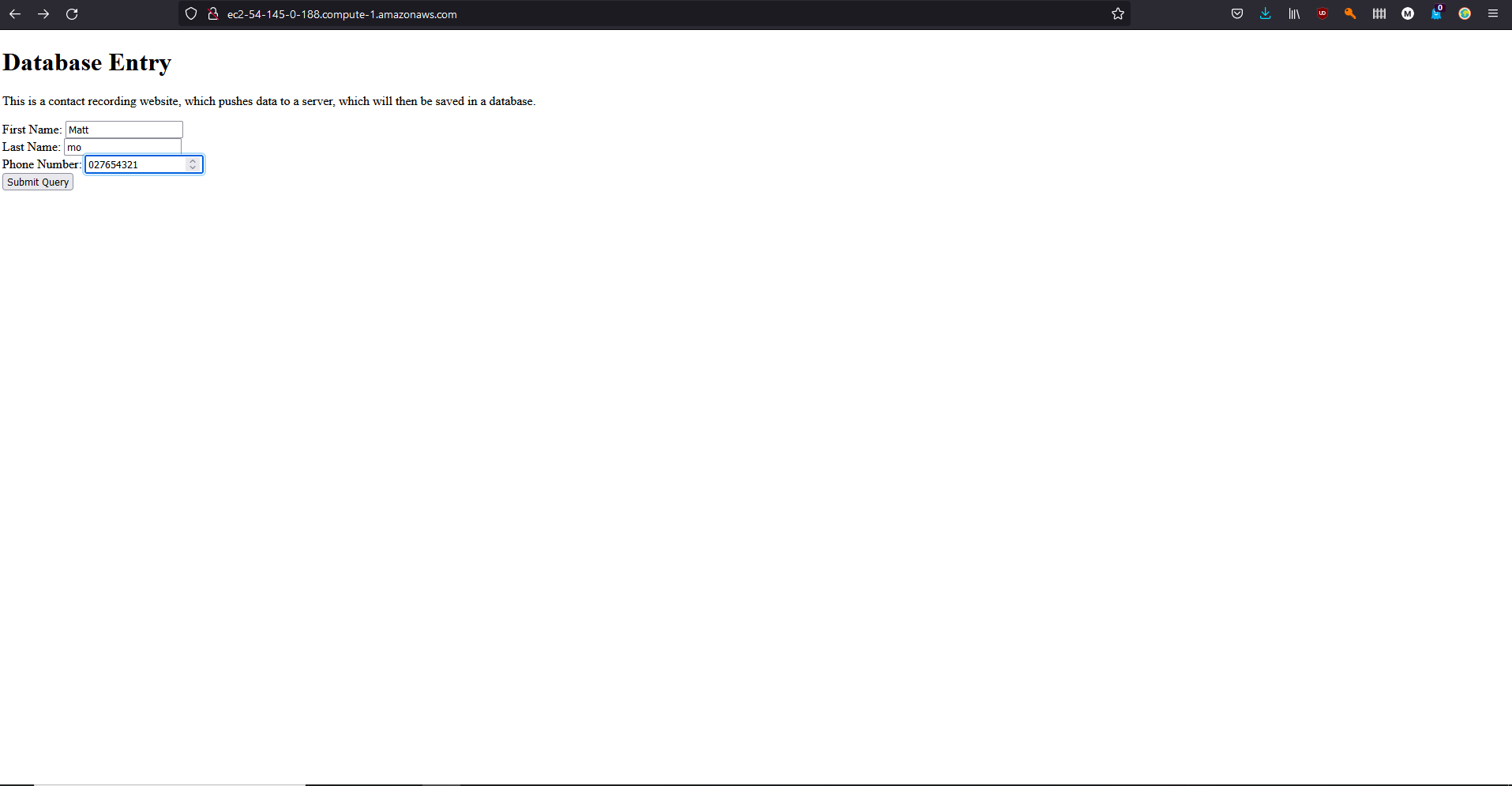
We have selected the 3 security groups to allow access from the webpages to the database and set the region to the same one we used for the EC2 instances.

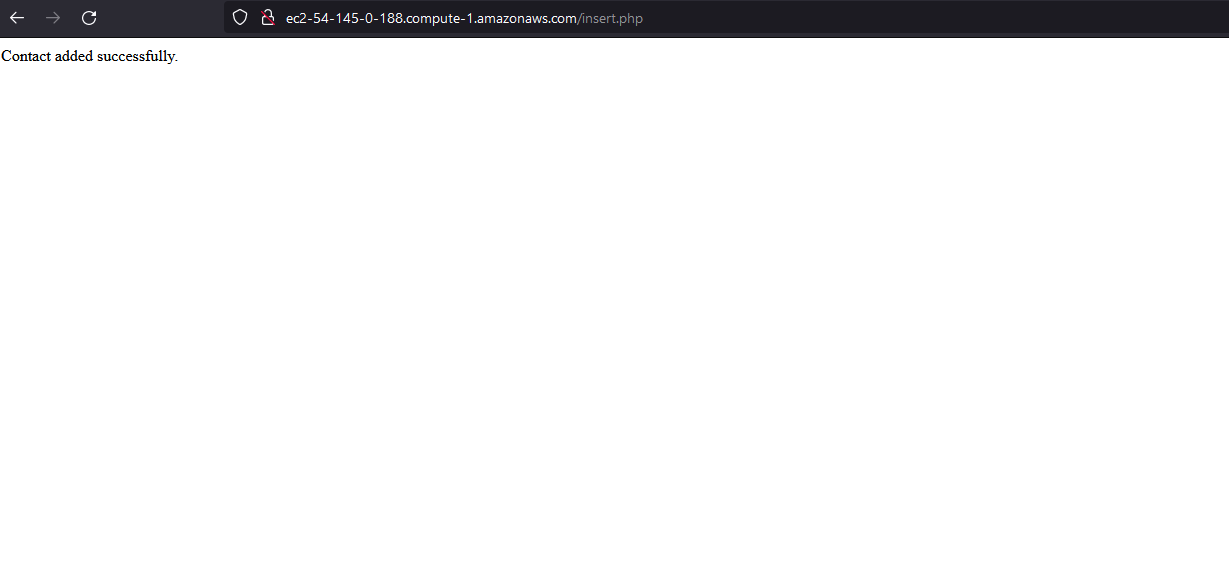
Only having password authentication as this service does not require a high degree of security. Thus, we prioritise ease of access.

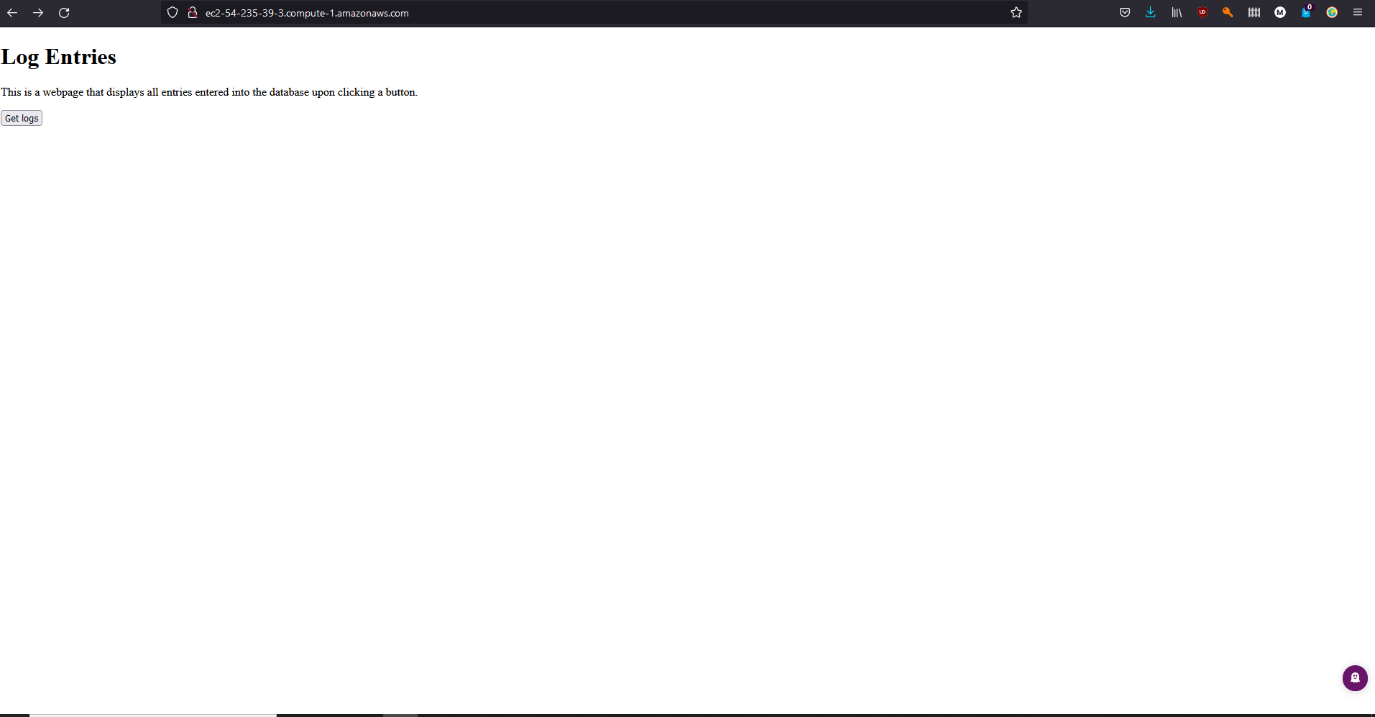
Enabling deletion protection to prevent an intern or new employee destroying the database and leaving minor version upgrades on, ensuring long term stability. Both are important for a long-term database.

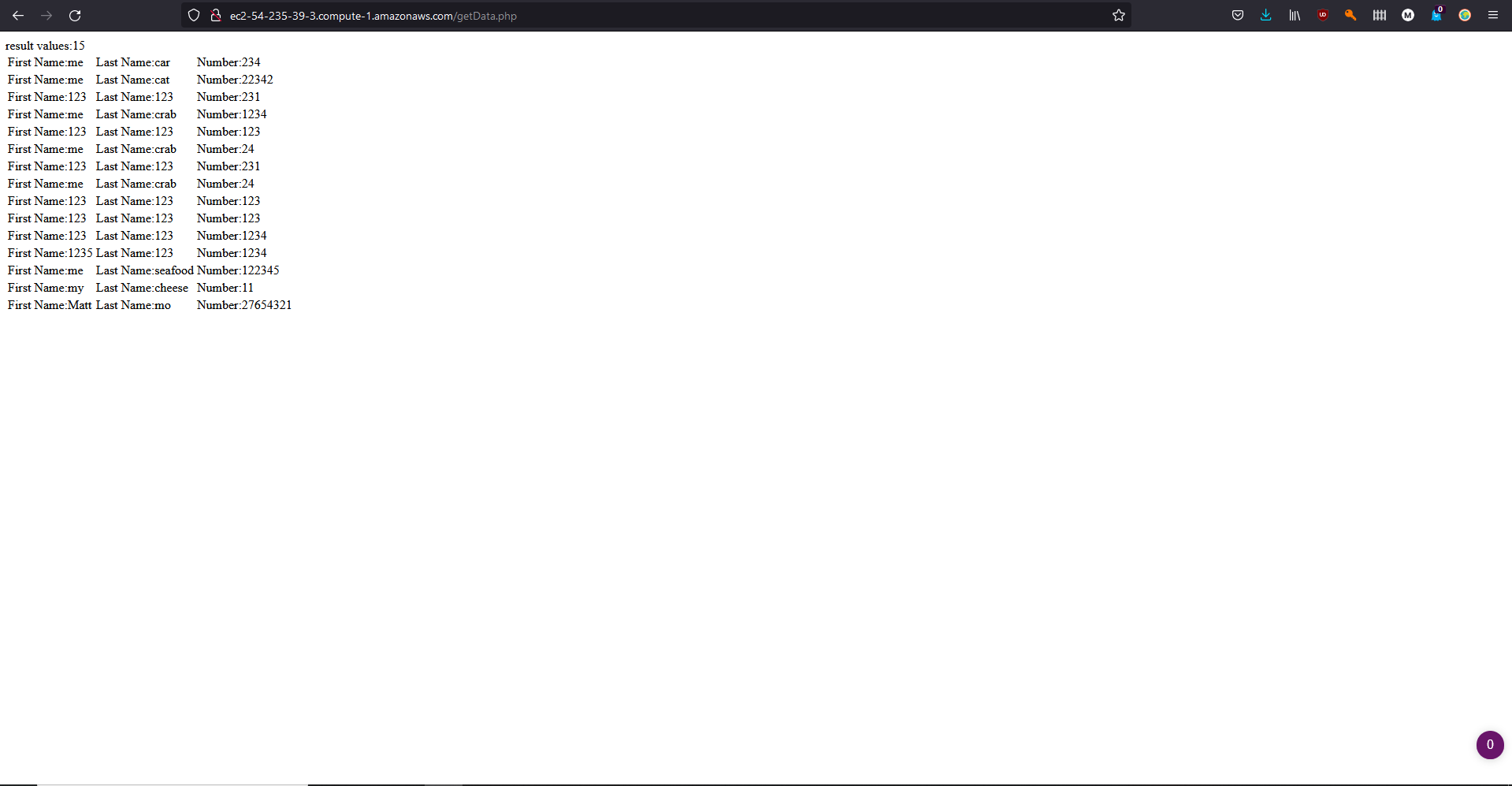


How to use our service

First input the contact info you want to record at @Matt, then click submit query.

If your data has been added, you will see the below screen.

Once you have some data, you can visit the next webpage at @Matt. When you have finished your data inputs, click Get Logs.

It will display the data in a neat table, along with how many entries there were.