**Case Study – Financial Report**

**“Only Flights”**

A picture containing text, clipart

Description automatically generated

Project Members & Student Numbers:

* Francisco Marcó (4467752)
* Sava Vasilev (4663438)
* Kaloyan Andreev (4408020)

Elaboration Date: 03/11/2022

Group 10

Table of content

[1. Preface 2](#_Toc122708508)

[1.1. Background 2](#_Toc122708509)

[1.2. Goal 2](#_Toc122708510)

[1.3. Expected result 2](#_Toc122708511)

[2. October 2022 Invoice Breakdown 3](#_Toc122708512)

[3. November 2022 Invoice Breakdown 6](#_Toc122708513)

[4. Preventive Actions taken 10](#_Toc122708514)

[3.1. Monthly Budget creation 10](#_Toc122708515)

[3.2. Stop inactive EC2 instances 14](#_Toc122708516)

[3.3. Stop inactive RDS instances 15](#_Toc122708517)

[5. Future Planning 16](#_Toc122708518)

[6. Figures 17](#_Toc122708519)

[7. Conclusion 19](#_Toc122708520)

# Preface

## Background

This document aims to provide an analysis and report of the financials of the Case Study Project. It had come to our attention that we, Group 10, have exceeded the given budget to work with on AWS. As so, we were tasked by our Tutor to develop this document to reflect on the finances of the project and outline a plan to follow so we can keep on continuing the development of our project.

## Goal

The main objective of preparing this analysis is to come up with an effective cost reduction strategy. To achieve this, we will make full use of the AWS provided tools to prepare reports, budget plans, alarms, and more.

## Expected result

By the end of this document, we will provide all the context for the spendings during the months of September – October, and November – December. Furthermore, we will take some extra steps to prevent excessive spendings and reduce costs in our infrastructure.

# October 2022 Invoice Breakdown

The following is the summary of the costs that our operation undertook during the month of October:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

We will go in detail over each Service, but firstly we discussed how the costs were distributed between each service. Naturally, the most essential components were racking up the most money (EC2 & VPC) but we were surprised to find other items in here that were more expensive than expected and others which we don’t know its origin.

Our first item in the invoice is pertaining to use of CloudWatch. We didn’t notice any anomalies in this field, as our use of CloudWatch has been limited so far. Because of this, we will keep an open eye to CloudWatch budgeting but it will not be a prioritized field (unless we start mismanaging our use and an intervention is needed to follow our budget).

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Next, we looked at one of the most crucial elements of our infrastructure: EC2 instances. We saw that most of the cost increase comes due to having the instances running on for too long. Although the number of hours shown isn’t exactly possible (clocking in 2000+ hours this month), we believe that there’s reason enough to consider a more considerate use of the instances. On the other hand, the EBS service has an adequate cost considering the usage we have given it, so we won’t be taken extra action for this field.



Imagen que contiene Gráfico de dispersión

Descripción generada automáticamenteThe following item is one that as of the writing of this document it is not clear its origin. We are currently being billed 50$, which is roughly a ¼ of our spendings during October, and after conducting light research we found that this service is normally used for registering Web Domains. Because we are not making use of one so far, we will look over this matter with our AWS account administrator, Gleb.

Next up, we looked at the Relational Database Service. Just like the EBS service, we didn’t find any major anomalies besides the hours spent using RDS. Because of that, we will aim to reduce the usage hours to reduce costs.



Lastly, we looked at the VPC field in the invoice. As it is one of our major components in our infrastructure, we discussed in-depth and decided that the main factor of cost is the hours we spend connected to the VPN and Transit Gateway VPN. For this, we aim to set a more effective work with these with the goal of being more cost effective.

That concludes the invoice of our spendings during October. As a conclusion, we decided to take preventive steps and better plan out our finances for the upcoming months. In the following sections, we will outline which actions we took and what is our finance planning.

# November 2022 Invoice Breakdown

The following is the summary of the costs that our operation undertook during the month of November. As a note, during this month the development of the MDP commenced, for which the same AWS account is being used as the Case Study project. Please take this into consideration while we go over the costs:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

We will go in detail over each Service, but firstly we discussed how the costs were distributed between each service. Like October’s invoice, we found that the more expensive items were key components of the projects, but we had one particular item which racked up a lot of the costs.

Imagen que contiene Aplicación

Descripción generada automáticamente

Firstly, we looked at the first item on the list which is CloudWatch. We believe that these costs are minimal, and they are the same as October’s invoice. In the future, we will be implementing an alternative monitoring tool (namely, Zabbix) which will help us reduce these costs even further.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Next up, we looked at one of the more important items: EC2 instances. Compared to October’s invoice, we managed to reduce costs by approximately 18%, which considering that we needed more resources than before because of the MDP, is a success and proves that the measures we implemented before worked to some extent. We believe that these costs could be reduced even further, with better and more structured planning. In the future, we will use what we are learning about costs management to implement it in our designs.

Imagen que contiene Gráfico

Descripción generada automáticamente

The following item in the invoice is the ECS. This is another crucial component of our projects, as we make use of it to implement CI/CD into our projects. Because we made use of Fargate instances, which charges you by vCPU hourly usage instead of hourly running time of the manage, we believe to have saved some money. However, we are not sure if these costs are expected for ECS or if they could be reduced even further by implementing policies into our Budget Thresholds.

Tabla

Descripción generada automáticamente

Moving on, we looked at Elastic Load Balancing. We saw that these costs were not very high, and it didn’t increase when comparing it to October’s invoice. As such, we will not be taking any actions regarding this item.



Something similar can be said about our next item, the Key management system. We did not find any anomalies in this field, as such we are not taking any further action at the moment.



Next, we looked at the Registrar. We successfully managed to configure Registrar for our Web Domain name and applied settings to reduce costs, which reduced them from 50$ to 11$. We consider this a major success as it has reduced our costs greatly.

Tabla

Descripción generada automáticamente con confianza media

This item proved to a major point of interest, as it is close to being ¼ of our monthly costs. The RDS instances were not configured properly, which caused them to be up and running for 7 days straight by accident and without our knowledge. This is an issue we are currently overlooking, and we will be implementing measure so this accident doesn’t happen again.

Tabla

Descripción generada automáticamente

Lastly, the final item in our invoice is Route53. As mentioned, we managed to setup our Web Domain which makes use of the Route53 service. Because these costs are minimal, we are not implementing any measures to reduce costs as of now.

This concludes the November 2022 invoice breakdown. All in all, we consider this month to be a success of the measures we took during the October 2022 financial analysis. However, we failed in implementing measures for the RDS instances (due to uncertainty about making use of RDS, which caused some slight confusion as we missed on implementing financial measures on it) which caused an unforeseen spike in our costs. Despite this, we reduced the overall costs by 30$, which considering that we are now working on a new project on top of the Case Study project shows that we succeed in implementing financial control measures.

# Preventive Actions taken

## Monthly Budget creation

One of the main ideas we took for having more efficient costs was to develop a Budget Plan by making use of the AWS Budgets tool.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamenteFirstly, we created an IAM role to allow AWS Budgets to manage our services according to the Budget Plan. We created the following role with these policies:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamenteAfter that, we created a Trial Budget Plan where we set our monthly spending limit to 80$, shown in the following image:

To make a more effective use of our budget plan, we set up different alarms which trigger according to our percentage in spending of our monthly limit. Furthermore, these alarms will trigger different actions in AWS which will limit the spending so we can’t go too overboard.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamenteThe first of these alarms’ triggers after our actual costs of the month reach the 85% of the budgeted amount. After this is achieved, the following action is triggered:

This action will alert to stop EC2 instances related to our Support, Admin, Chat Server, and Monitoring server. Because these are additional functionalities to our solution, we can reduce costs by terminating unneeded resources. This action however will only alert to stop the instances, as it could happen that crucial work is being carried there and the machine needs to be up. Likewise, if it is necessary to run the machines again even after our budget costs are over 85% it is possible to do so.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteNext, we set up an alarm for when our forecasted costs reach 100% of our budget plan.

To note, this alert triggers no action in AWS. It will only notify us and serves the purpose of alerting us when we may be reaching our spending limit if we maintain the same spending trend.

Finally, the final alert we created is triggered when we reach 100% of our Budget Plan in our actual spending. After that, the following action is triggered:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Texto

Descripción generada automáticamenteThis action will enforce a custom created “Limit-EC2” policy unto the development team of the project. This policy enforces the following:

This policy denies us of interacting with the EC2 instances. This will greatly reduce costs as we won’t be charged for EC2 after exceeding our spending limit. It is a more drastic last resort, but as we start drafting our financial plan we may come up with something more efficient.

To have a clearer look at our newly created budget plan, we configured to receive weekly Budget Report every Monday. This is created through AWS and will be uploaded to the git repository as we receive them.

## Stop inactive EC2 instances

Tabla

Descripción generada automáticamenteAs an additional measure, we decided to create Alarms for some of the EC2 instances in our infrastructure. These will be ongoing and detect if an instance has been inactive for more than an hour. If that is the case, the instance will be stopped. This will reduce our spendings as we will not encounter any problems if we forget to stop a running instance.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamenteThis alarm is not applied to instances that may need constant running, such as the web server and chat server. As a further look, this is the thresholds that the alarm follows:

When the CPU utilization falls under .2%, which is essentially idle, the Alarm is triggered, and the instance is stopped.

## 3.3. Stop inactive RDS instances

As mentioned on the November 2022 invoice breakdown, we had an issue with the RDS instances as they were left running unattended for an extended period. We originally planned on implementing measures to prevent this, but it was uncertain whether we were going to make use of this service. As we kicked the decision down the road, we forgot about implementing financial control measures.

Moving on, after the Winter Break, we will be implementing alerts and alarms which will turn off these instances when they are not being used. Additionally, we plan to be more responsible with these resources as we wrap up the semester.

# Future Planning

Moving on with the projects, as we approach the end of the semester, we plan on taking some final measures to reduce costs and to prove that we are taking this matter into serious consideration. We will be more responsible during this last stage with the AWS resources.

Some of the tentative measures we will be taking are the following:

* Making use of the AWS Cost Anomaly Prevention tool
* Shutting down all instances when they are not being used1
* Optimizing our resource management when working on the MDP project.
* Monthly budget re-iteration, with an increased 200$ monthly limit
* Making a Cost and Usage report, with the help of our AWS account manager Gleb as it is not possible for us to generate one because we’re not the paying account

As we start implementing some of these or other measures, we will include them in the “Preventive actions taken” chapter of this document.

# Figures

Gráfico, Gráfico de barras

Descripción generada automáticamente

Image 1: Bar graph of costs by service type during the months of October and November.

Gráfico

Descripción generada automáticamente

Image 2: Bar graph of daily costs by service type, from November 1st to December 21st

Gráfico, Gráfico de líneas

Descripción generada automáticamente

Image 3: On the left side, a Line graph of services usage by cost. On the right side, a table detailing services, its average cost, and Group 10’s November total by service.

# Conclusion

We believe to have been learning a lot in financial management by making this report. While sometimes it’s regrettable that we are not being efficient in our Service usage in AWS, we are starting to feel more confident and comfortable with dealing with the financial side of Cloud Computing. At the end of the day, one of the more important selling points of Cloud Platforms is cost efficiency, and if in the future we want to work with these technologies we have to learn how to effectively demonstrate this. Because if not, why would a client migrate to the Cloud when the same thing can be achieved in a cheaper way by using on-premises resources?

As previously mentioned, moving on we will take more measures to be more efficient in our Costs Management. We also would like to thank our Teachers for not “pulling the plug” or interfering at any moment and allowing us to reflect and learn on our own about financial management in AWS.