City, University of London

BSc Computer Science with Placement year

Auditing tool for AWS machine instances and images

*Short description here*

By Irina-Maria Fratila [irina-maria.fratila@city.ac.uk](mailto:irina-maria.fratila@city.ac.uk)

Consultant name: Dr. Laure Daviaud

Table of Contents

[Chapter 1: Introduction 3](#_Toc129179891)

[Chapter 2: Output summary 3](#_Toc129179892)

[Chapter 3: Literature review 3](#_Toc129179893)

[3.1. Cloud computing benefits and knock-on-effects on enterprise operations 3](#_Toc129179894)

[3.2. Technology and services to be used 4](#_Toc129179895)

[3.3. Similar applications on the market 5](#_Toc129179896)

[Chapter 4: Method 6](#_Toc129179897)

[4.1 Software development methodology 6](#_Toc129179898)

[4.2 Work plan 7](#_Toc129179899)

[4.2.1 Initial plan 7](#_Toc129179900)

[4.2.2 Application of work plan and adjustments made to project/plan 8](#_Toc129179901)

[4.3 Requirements documentation and tracking completeness 8](#_Toc129179902)

[4.4 Documentation of design/architecture 8](#_Toc129179903)

[4.5 Implementation of system components and version control 8](#_Toc129179904)

[4.5.1 Development 8](#_Toc129179905)

[4.5.2 Version control 8](#_Toc129179906)

[4.6 Testing 9](#_Toc129179907)

[Chapter 5: Results 9](#_Toc129179908)

[Chapter 6: Conclusions and discussions 9](#_Toc129179909)

[Reference list 9](#_Toc129179910)

[Appendices 9](#_Toc129179911)

# Chapter 1: Introduction

# Chapter 2: Output summary

# Chapter 3: Literature review

The following chapter references various literature to understand the context in which the auditing tool will operate. It includes a thorough review of the cloud computing industry, what benefits does it offer to clients and its knock-on-effects on business operations. I also specifically explore Amazon Web Services, how have containers become key in the deployment of reliable services and why use a tool like Kubernetes to manage those containers. Finally, I briefly cover other existing alternatives to my tool and why they may not be feasible to my client.

## 3.1. Cloud computing benefits and knock-on-effects on enterprise operations

Cloud computing is the on-demand availability of computing power, data storage and various other services. It typically uses a “pay as you go” model where the user only pays for what resources they use, often reducing capital costs.

When discussing cloud computing benefits, I will look at the main elements organisations make in comparison to their existing IT resources. Venters and Whitley suggest that there are five categories vendors look at when considering cloud computing: security, availability and latency equivalence to current infrastructure, variety in services, abstraction of maintenance and scalability (Venters and Whitley, 2012).

Equivalence is the first criterion cloud computing has to satisfy to be considered by clients. Security of data, availability of their servers and response times must at least be as good as on-premise IT infrastructure. Variety refers to the wide range of services made available by a cloud provider and their flexibility, for example, Amazon Web Services offer 200 services across 31 regions, from California in the USA to South Korea’s Seoul. (AWS, n.d.).

Abstraction in cloud computing is to hide the complexities of managing IT infrastructure and software from the user. This idea lies on a spectrum, from AWS’s EC2 instances where the user has access to a virtual machine and its computing power to run any software, to AWS’s Lambda service. Labda is a serverless computing service where the user uploads the source code and names triggers, AWS takes care of all physical and software infrastructure, only running the code in response to the specified triggers. The abstraction of managing physical hardware, power and cooling, backups of data, redundancy, trained staff required and security of the building itself are significant factors for cost savings(Venters and Whitley, 2012).

The next category is scalability, which also ties into the “pay as you go” model. This feature is the heart of cloud computing, the ability to rapidly increase or decrease computing power in response to demand, only paying for what you use. A problem identified by Venters and Whitley in an interview with an IT service director was the over-provisioning of IT hardware, “they look at a project and implement enough IT infrastructure to support that project for up to three years. And typically you might find that only 50% of that capacity is used up until the third year of the project. It’s spent two years sitting around doing nothing” (Venters and Whitley, 2012). Cloud computing, more specifically the elastic cloud eliminates this problem severely, making computing power easy to access and scale. The upfront cost of developing an application or service is significantly lower, allowing smaller enterprises and governments with limited capital to succeed and develop reliable products.

Overall, cloud computing has given businesses from start-ups to corporations a better way to manage and deploy their products. Scalable computing power, lack of hardware with associated power and cooling costs, and reduced number of IT staff are just some of the benefits that have completely changed the business world.

## 3.2. Technology and services to be used

A large variety of software is to be used during development of the auditing tool; therefore their relationship has to be carefully studied and implemented. While the relationship with my client is informal, there are technologies I must use during development so that their engineering team can develop this application further. In this subsection I will cover AWS services, containerisation, container, and infrastructure management.

The high-level goal of my web application is to easily observe which AWS resources are older than “x” number of days, therefore it is required that I have fundamental knowledge of AWS and their most popular services. I undertook the online course “AWS Cloud Technical Essentials” offered by AWS (AWS, n.d.). The course covered best practices regarding account security, identity and access management, different types of computing, storage and database services and various ways of monitoring your AWS resources. I will expand further on each section in “Chapter 4” as to what service I chose for a particular application requirement.

One of the required methodologies is deploying the web application to containers using Docker and managing those containers with Kubernetes. Containers are a packet of software which contain the source code and all their dependencies. It is a standalone unit which can be seamlessly run and deployed to different machines or environments (Docker, n.d.). Kubenetes is a platform for orchestrating containers, this includes automating services, scaling, and management of complex containerised workloads in the cloud (Kubernetes, n.d.). When developing the application, I will carefully study best practices about creating the relationship between the containers and Kubernetes in an AWS cloud environment. I plan on working through “Containerized applications on AWS” course offered by AWS themselves to study and learn good engineering practices (AWS, n.d.).

## 3.3. Similar applications on the market

When looking at similar products on the market there are two options, using the AWS console to perform checks manually, or implement a third-party application that manages all cloud resources under an account.

One can use the AWS management console web page to get all the information required about an EC2 instance and respective AMI, including the date it was launched. This manual task is not scalable as a large enterprise will likely have dozens of instances with various critical services running, as well as multiple IAM users under the root account. Many resources and users create room for human error, therefore not making this a viable solution for a large enterprise.

Third-party applications like CloudCheckr, CloudHealth, DataDog and many more, offer a cloud management service that covers cost optimisation, automation, performance, and security (CloudZero, 2023). These services cover many features, including my tool, getting the age and health of EC2 instances under an account. While a service like this may be useful for small to medium enterprises with smaller capital, my client is in need of a specific auditing tool, suggesting that a “do it all” product like this is not necessary.

Graphical user interface, text, application, email

Description automatically generated

Figure 1 Screenshot of the AWS management console when looking at an EC2 instance's information in thorough detail. In red is boxed the date of the instance’s launch. The instance “employee-directory” has been running for approximately 8 days.

# Chapter 4: Method

## 4.1 Software development methodology

What method did I use?

What is this method, explained briefly?

Why is it a good fit for my project and any changes I made to suit my needs.

How did I apply and use it on my development cycle?

During the software development a Test Driven Development approach was taken. This methodology relies on software requirements to be converted to test cases before their implementation, checking the completeness of a feature against the test cases created (Wesley, 2002). The technical requirements for the auditing tool are partially defined by my placement company and myself, but because our relationship is informal there are no stakeholders to verify my output during development. Therefore, this methodology seemed to be the best fit for the development process. A feature is taken from the work plan, build a test for it, reiterate over the script until test is passed. Once the pest is passed, the feature can be marked as complete and the process repeats.

In terms of time management, I will scrum sprints of two weeks in length as discussed in the Project Development Plan. The project requirements are potentially split to smaller tasks and assigned a sprint, where at the end of each sprint there is a team meeting (sprint retrospect) to reflect on the progress of the tasks on that specific sprint. I am working alone, so at the end of each sprint I will assess my progress on the tasks and adjust them as needed. I believe this methodology is suitable for my project as it will allow for consistent output and reflection on my development progress, while it also being widely adopted in the industry and improving my professional skills. In the next section, I will go in greater detail about the work plan, how I stayed on track, and any adjustments I made during development.

## 4.2 Work plan

Intro

### 4.2.1 Initial plan

As discussed above and in the project proposal, I will use an agile scrum methodology to manage and deliver my project. In general, each sprint is typically 2 weeks long and is made up of four stages: sprint planning, daily scrum, sprint review and sprint retrospective. The sprint plan I created initially with the application Notion is shown in Figure 1, where the project requirements are split into smaller tasks and added to sprints. I used my best judgement to estimate how long a task will take, taking in consideration what skills I already know. There are exactly seven sprints, where the final sprint is just a week long, coinciding with the week between the code and report submission.

Table

Description automatically generated

Figure 1 All tasks created for the project as of 5 Feb 2023, organised in sprints

In terms of the daily scrum, sprint review and sprint retrospect, those must be adjusted slightly to suit a one-person team like myself. The daily scrum is a daily meeting where a team briefly goes through their progress in the context of the whole sprint, and for myself, at the start of each working day I reflected on the plan and decided what is to be completed that day. Sprint review and retrospect is combined to one activity for me, and it was an opportunity for me to reflect on my pace of work or make any adjustments to the plan. In the next subsection I will go through each sprint of the project, with their reflections and potential changes to the project/plan.

Touching on the length of a sprint, I chose two weeks, but the official guide suggests any amount between one and four weeks (scrumguides.org, 2020). Based on my previous work experience and knowledge of my workflow, two weeks seemed to be the most optimal length. It allowed me to frequently reflect on the previous sprint, what I could improve and continue doing. The reflections of each sprint are discussed below.

### 4.2.2 Application of work plan and adjustments made to project/plan

In this subsection I will present the results of each sprint, how well I was able to stay on track, and any adjustments made to the plan.

Sprint 1 & 2: When creating the initial plan, I did not consider studying for literature review, so that was added in Sprint 1, and the development setup was pushed to Sprint 2. As a result, in Sprint 2 I successfully created the AWS Virtual Private Cloud network needed for the development of the auditing tool, as well as creating computing instances and images needed for testing in the network. I stayed on track with the workload and finished tasks within the expected timeframe.

Sprint 3: finished all the coding within the week, just did writing for the rest. Ive started a bit of sprint 4

## 4.3 Requirements documentation and tracking completeness

## 4.4 Documentation of design/architecture

## 4.5 Implementation of system components and version control

### 4.5.1 Development

#### 4.5.1.1 Backend data collection job (Python & boto3)

#### 4.5.1.2 Front end web application

#### 4.5.1.3 Containerisation with Docker

#### 4.5.1.4 Container management with Kubernetes

#### 4.5.1.5 Infrastructure management with Terraform

### 4.5.2 Version control

## 4.6 Testing

# Chapter 5: Results

# Chapter 6: Conclusions and discussions

# Reference list

AWS, n.d. *AWS Cloud Technical Essentials.* [Online]   
Available at: https://www.edx.org/course/aws-cloud-technical-essentials

AWS, n.d. *AWS Global Infrastructure.* [Online]   
Available at: https://aws.amazon.com/about-aws/global-infrastructure/#:~:text=The%20AWS%20Cloud%20spans%2099,%2C%20New%20Zealand%2C%20and%20Thailand.

AWS, n.d. *Containerized Applications on AWS.* [Online]   
Available at: edx.org/course/containerized-applications-on-aws?index=product&queryID=1ce4f8f86ecb2bb04ef89e4abfd5fad0&position=4

CloudZero, 2023. *CloudHealth Vs. CloudCheckr (And Is There A Better Option?).* [Online]   
Available at: https://www.cloudzero.com/blog/cloudhealth-vs-cloudcheckr

Docker, n.d. *Use containers to Build, Share and Run your applications.* [Online]   
Available at: https://www.docker.com/resources/what-container/

Kubernetes, n.d. *Overview.* [Online]   
Available at: https://kubernetes.io/docs/concepts/overview/

scrumguides.org, 2020. *The 2020 Scrum Guide.* [Online]   
Available at: https://scrumguides.org/scrum-guide.html#the-sprint

Venters, W., 2012. *A Critical Review of Cloud Computing: Researching Desires and Realities.* [Online]   
Available at: https://doi.org/10.1057/jit.2012.17

Wesley, A., 2002. *Test-Driven Development by Example.* s.l.:s.n.

Whitley, E. A., 2012. *A critical review of cloud computing: researching desires and realities.* [Online]   
Available at: https://doi.org/10.1057/jit.2012.17

# Appendices