Cloud and Computer Architecture – Assignment 2

Deploying a Microservice on AWS/Azure

Contents

[Background 2](#_Toc185705945)

[Purpose 2](#_Toc185705946)

[Discussion 2](#_Toc185705947)

[Cloud Services](#_Cloud_Services)

[Microservices](#_Microservices)

[Setting Up a Cloud Account and Instance](#_Setting_Up_a)

[Using Linux Commands During Deployment](#_Using_Linux_Commands)

[Installation of Java and RDBMS](#_Installation_of_Java)

[Cloud Storage on AWS](#_Cloud_Storage_on)

[Bibliography 16](#_Bibliography)

## Background

The term “Cloud Services” describes delivery of a wide range of computing services provided across the internet such as storage, processing resources, databases, networking, and software tools. These services are provided by Cloud service providers, such as Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform, and are typically billed on a pay-as-you-go basis.

## Purpose

The purpose of this report is to summarize the activities undertaken to demonstrate an understanding of Cloud Services and Microservices by deploying a simple microservice on a Cloud platform. The report will provide a detailed explanation of the processes involved in setting up and configuring a virtual system instance, along with the steps taken to add required functionality which enables routine operations to be performed using the system.

## Discussion

#### Cloud Services

Cloud services play a crucial role in modern computing for several key reasons:

Cost-effectiveness:

* Reduces investment requirements for hardware and software.
* Operates on a pay-as-you-go model, ensuring businesses only pay for what they use.

Scalability and Flexibility:

* Resources can be scaled to suit business demand.
* Ideal for businesses with varying workloads and priorities.

Security:

* Cloud providers utilise effective security measures, including access controls and encryption of data at rest and in transit. Compliance to international standards (e.g., ISO27001, SOC 2) is usually achieved prior to operation of services.

Rapid Deployment:

* Cloud providers offer Platform as a Service (PaaS) environments out-of-the-box, for faster development and deployment.

Reliability and Availability:

* Cloud providers ensure high availability of client data through redundancy, failover systems, and distributed data centres.
* Disaster recovery and backup solutions are often built-in to the deployed Cloud resources.

**Benefits of Cloud Computing**

Accessibility and Mobility

* Cloud computing allows users access to their data wherever they have access to an internet connection and appropriate device. This ensures that employees, clients, and customers have current information.

Centralised Security

* Centralising data backups reduces data loss risk and allows for data restoration in the event of a failure or disaster.

Reliability

* Providers ensure 24/7 reliability though utilisation of redundancy system structures and associated software.

Disaster Recovery and Business Continuity

* Store configurations for critical data and applications prevents data loss from hardware malfunctions, natural disasters, or other unforeseen circumstances.

**Challenges of Cloud Computing**

Data Security and Privacy

* Protecting sensitive data on the cloud requires robust authentication, encryption, and access controls. Threats like data breaches, identity theft, and malware can erode user trust and harm business reputation. High-speed data transfers also increase the risk of leaks.

Performance Issues

* Latency, inefficient load balancing, and fault tolerance issues can negatively affect user experience. System optimisation is essential to maintain consistent performance.

Network Dependence

* Cloud operations rely heavily on stable, high-speed internet connections. Limitations to bandwidth or connection outages can disrupt routine business operations, particularly for smaller companies.

Learning Curve

* Training may be required to ensure that users adapt to cloud platforms, especially if advanced tools and services are required.

#### Microservices

Applications built using microservices consist of small, independent services that communicate through well-defined APIs, enabling a modular and flexible approach to software development. Each service is designed to perform a specific function, and they are loosely coupled, allowing for independent development, deployment, and scaling.

Microservices architecture is important because it enables faster releases, greater agility, scalability, and improved team productivity. By decoupling services, businesses can respond to changing demands quickly and maintain routine operations efficiently. This architecture not only addresses the challenges of existing monolithic systems but also allows organisations to build modern, adaptable solutions that support future development.

#### Setting Up a Cloud Account and Instance

An instance represents a virtualised computing resource that operates as if it were a standalone physical machine. It provides scalable computing power, allowing users to run applications, host websites, perform data processing, or execute other tasks typically performed by physical servers. Instances are based on a core set of compute, storage, database, and networking services. Instances provide a versatile platform for running web applications, APIs, data processing models, and high-performance computing tasks like simulations and 3D rendering.

The importance of instances lies in their scalability, cost efficiency, and flexibility of service. Using virtual instances reduces necessity for the physical infrastructure usually required for routine business requirements or continuous computational research.

**Setting up an Amazon Web Services (AWS) account**

For the purposes of this project an AWS Free Tier account was set up. This involved assigning an email address, AWS Account Name, user password, contact information and payment method.

Multifactor Authentication (MFA) can also be configured but this is not necessary immediately after setup of an account.

Once an account is established, it is possible to configure instances of a virtual system.

**Setting up an instance of a virtual machine**

After logging into AWS with a valid account, the user can access the Amazon Elastic Compute Cloud (EC2) Dashboard.

A screenshot of a computer

Description automatically generated

From here, it is possible to configure virtual machines, instances of virtual machines, storage, configure monitoring and metrics, and security settings including management of Network ACLs (NACLs) through the Amazon VPC (Virtual Private Cloud).

Selection of “Launch Instances” allows the user to configure a new instance before launching. Instance name and / or resource tags can be assigned e.g. “Web Server - Production”, “Database-Dev”.

AWS has a catalogue of Operating System (OS) images and Applications. More widely used OS options like Windows, MacOS and Ubuntu are available under “Quick Start” including a distribution of Linux developed specifically for AWS: Amazon Linux. Amazon Linux 2023 (64-Bit, x86) was initially selected for this project.

A screenshot of a computer

Description automatically generated

A key pair was set up for use with OpenSSH (.pem file) as the proposal was to use a separate external Linux distribution (MXLinux) with built-in OpenSSH functionality to access the AWS instance.

A screenshot of a computer

Description automatically generated

Storage was configured with a default value of 8GB

A screenshot of a computer

Description automatically generated

After the instance set up was successfully completed, it was launched within the AWS Web portal. From the EC2 dashboard, I selected the new instance and connected to it.

A screenshot of a computer

Description automatically generated

#### Using Linux Commands During Deployment

Once connected to the new instance using Amazon Linux, I ran a “top” command to confirm that processes were running on the instance.

A screenshot of a computer

Description automatically generated

Due to personal preference and ease of use, I choose to use an external Linux distribution (MXLinux) set up through a hosted hypervisor (VirtualBox) to access the new AWS instance.

This required setup of an Inbound Rule to use SSH allowing access from my IP only.

A screenshot of a computer

Description automatically generated

I then set up a connection to the running instance using Secure Shell (SSH).

A computer screen shot of a computer

Description automatically generated

#### 

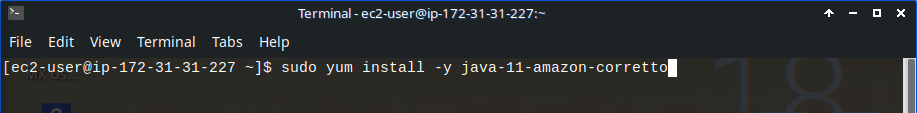
#### Installation of Java and RDBMS

**Java (Amazon Corretto 11)**

Note: ‘sudo’ is used throughout installation to elevate privileges which allow installation to the Amazon Linux configuration. ‘sudo su’ can alternatively be used to set user as ‘root’ with superuser permissions.

To install the Amazon Linux-compatible version of Java (Amazon Corretto 11), a *yum* command was used:

*sudo yum install -y java-11-amazon-corretto*



After installation, the Java version was confirmed as 11.0.25.9.1

A screenshot of a computer

Description automatically generated

**MariaDB Database Relational Management System (RDBMS)**

The current version of MariaDB on the default AWS Linux 2 repository was installed using command

*sudo yum install -y mariadb105-server*

A screenshot of a computer

Description automatically generated

The following commands were used to initialise the application:

*sudo systemctl enable mariadb*

*sudo systemctl start mariadb*

Status of the MariaDB service was checked using command

*systemctl status mariadb*

A computer screen shot of a program

Description automatically generated

A password for the root user for MariaDB was set up was set up using command

*sudo mysql -uroot -p*

This initialised the MariaDB application.

**Database Creation**

After initialising MariaDB, SQL commands can be used to modify and manipulate databases.

There are databases configured in MariaDB by default. These can be viewed using command

*show databases;*

A new database “projectdb” was created using command

*create database projectdb;*

This database was displayed in list of databases after creation.

A screen shot of a computer

Description automatically generated

*use database projectdb;* was used to set ‘projectdb’ as the active database. A new table “userdetails” was then created in this database using SQL commands:

*create table userdetails (*

*userId INT AUTO\_INCREMENT PRIMARY KEY,*

*code VARCHAR(20) UNIQUE,*

*userName VARCHAR(50),*

*userAge INT,*

*userLocation VARCHAR(80)*

*);*

A list of tables can be viewed using command

*show tables;*

or *describe userdetails;* to get detailed table information.

A screen shot of a computer

Description automatically generated

Due to the complexity of creating or modifying an application, a new application was not created for this project to test utilisation of a Microservice. A database (servicedb) and table (coupon) was set up to meet the configuration of an existing application which will be used for this project (couponservice-0.0.1).

The couponservice application maps "/couponapi" as the base URL.

The create() method maps to "/coupons" and handles HTTP POST requests. It accepts a Coupon object from the request body and saves it to the database using repo.save(coupon).

The getCoupon() method maps to "/coupons/{code}" and handles HTTP GET requests. It retrieves a coupon from the database by querying the code parameter using repo.findByCode(code).

Default values for URL, mySQL username and password, and port number for server requests are hardcoded in the application code.

A screenshot of a computer program

Description automatically generated

The “servicedb” database was created in the new AWS instance and a “coupon” table was set up for use.

*create table coupon(*

*id INT AUTO\_INCREMENT PRIMARY KEY,*

*code VARCHAR(20) UNIQUE,*

*discount DECIMAL(8,3),*

*exp\_date VARCHAR(100)*

*);*

A screen shot of a computer

Description automatically generated

#### Cloud Storage on AWS

AWS offer a secure object storage service (Simple Storage Service / S3) which can be accessed from the AWS Management Console when a user selects “S3”

A screenshot of a computer

Description automatically generated

From the S3 page, a new storage bucket was created for secure file storage using “Create bucket”

A screenshot of a computer

Description automatically generated

Once a name for the Bucket was entered, all default values were used.

Bucket Type: General purpose

Object Ownership: ACLs disabled.

Block all public access.

“Create bucket” was selected to create a new bucket called “project2024bucket”.

After selecting this bucket from list of General purpose buckets, files for a basic website were uploaded from a local directory using Upload option.



A screenshot of a computer

Description automatically generated

These files could then be viewed, selected, and opened from within the bucket.

The couponservice application (.jar file) from a previous Moodle lab session was uploaded to “project2024bucket” so that use of a Microservice could be demonstrated.

A screenshot of a computer

Description automatically generated

As access settings were previously set using default values during creation of the Bucket, it was necessary to remove the Block on public access and set permissions to allow for files to be accessed outside of the S3 environment.

“Block public access (bucket settings)” were altered to remove “Block all public access” and the following Bucket policy was used:

*{*

*"Version": "2012-10-17",*

*"Statement": [*

*{*

*"Effect": "Allow",*

*"Principal": "\*",*

*"Action": "s3:GetObject",*

*"Resource": "arn:aws:s3:::project2024bucket/\*"*

*}*

*]*

*}*

An additional Inbound Rule was added through EC2 > Security Groups to allow Port 9091 to be used as this is required for HTTP Requests using the couponservice application.

A screenshot of a computer

Description automatically generated

The AWS instance was rebooted using *sudo reboot*.

To run the application in the instance of Amazon Linux, the *wget* command was used to download the .jar file from the AWS S3 storage bucket so that it could executed in Linux.

*wget* [*https://project2024bucket.s3.eu-west-1.amazonaws.com/couponservice-0.0.1-SNAPSHOT.jar*](https://project2024bucket.s3.eu-west-1.amazonaws.com/couponservice-0.0.1-SNAPSHOT.jar)

The couponservice application was executed using command

*java -jar couponservice-0.0.1-SNAPSHOT.jar*

A screen shot of a computer

Description automatically generated

The following command was used to confirm that couponservice was active

*ps aux | grep couponservice*

To set up the couponservice to run automatically on instance startup, an *rc.local* file was created. This file was edited using *vi etc/rc.local* to include a call to the couponservice application on startup on an instance.

A screenshot of a computer

Description automatically generated

The file contents can be viewed / verified using the *cat* command.

A screen shot of a computer

Description automatically generated

To allow execute permissions on the etc/rc.local file, the following was applied

*sudo chmod +x /etc/rc.local*

**Sending data to the database**

Postman (<https://web.postman.co>) was used for sending data to the created database – projectdb.

A new POST request was set up on the Postman web platform in the form of body (e.g., JSON) data to test sending of data to the projectdb database to http://54.224.83.112:9091/couponapi/coupons.

{

  "code": "testCoupon",

  "discount": 17.5,

  "expDate": "12/12/2024"

}

A screenshot of a coupon code

Description automatically generated

A GET command request using the ‘code’ parameter “testCoupon” <http://54.224.83.112:9091/couponapi/coupons/testCoupon> returns the posted data with id ‘1’ assigned.

A screenshot of a computer

Description automatically generated

## Bibliography

Citrix.com. (2024). *What is a Cloud Service? – Cloud Services Solutions - Citrix*. [online] Available at: <https://www.citrix.com/glossary/what-is-a-cloud-service.html?srsltid=AfmBOorNqMA4gVJZmk5C9RnDnbJkboVOP2ghZeFKsqCUChceAl_sRXkn>

Quora.com. (2024).  [online] Available at: <https://www.quora.com/What-role-does-cloud-computing-play-in-modern-data-centers>

ITarian. (n.d.). *How to use Remote Desktop*. [online] Available at: <https://www.itarian.com/remote-desktop.php>

‌

Hivenet.com. (2024). *The Importance of Cloud Computing: Why it Matters for Businesses in 2024 | Hive*. [online] Available at: <https://www.hivenet.com/post/importance-of-cloud-computing-why-it-matters>

TUS - Computer Architecture & Cloud Computing Lecture Notes. Available at: <https://moodle.midlands.tus.ie/course/view.php?id=19012>

Wikipedia Contributors (2019). *Microservices*. [online] Wikipedia. Available at: <https://en.wikipedia.org/wiki/Microservices>

‌

‌