Secure and Scalable AWS Cloud Infrastructure

Group Project Assignment

Submitted By: Group 4

Group Members: Vanshika, Gurkaran Singh, Gulab Singh and Prarthana Gopalkrishna Patel

Submitted To: Nikola Cedic

Course Code: INFO 2350

Submission Date: August 05, 2025

# Table of Contents

[Table of Contents 2](#_Toc205336918)

[Group Project – Designing a secure and scalable AWS Cloud Infrastructure 3](#_Toc205336919)

[Project Overview 3](#_Toc205336920)

[Group Members Roles and Project Method 3](#_Toc205336921)

[Task 1: Architecture Diagram 4](#_Toc205336922)

[Task 2: Implementation 5](#_Toc205336923)

[Task 3: Automation 13](#_Toc205336924)

[Task 4: Security Considerations 15](#_Toc205336925)

[References 15](#_Toc205336926)

# Group Project – Designing a secure and scalable AWS Cloud Infrastructure

# Project Overview

The aim of the project is to design, develop and deploy a secure and scalable cloud-based infrastructure on Amazon Web Services (AWS) for a simple web application which interacts with a backend database.

Key Features of this Infrastructure are as follows:

* Custom VPC Design: configured with the web application deployed in public subnet and backend database server deployed in the private subnet.
* Network: Proper network is build using routing tables, NAT and Internet Gateway.
* Secure Communication: Connectivity between the web and database is secured using routing protocols and security groups. The purpose of this is to allow only internet access to the web application for public and restrict public access to databases.
* Best Practices: use of ssh key pair for access, security groups for inbound rules and using least-privilege access method to prevent access.
* Automation: Using Infrastructure as Code tools to recreate the cloud-based solutions with the least number of steps.

# Group Members Roles and Project Method

For this project, we chose the method of using Learner’s lab. We first did the implementation on Gurkaran’s AWS and then followed the same on the rest of the members AWS.

Though we implemented the lab together, for better management, we divided the tasks to save time. Following describes the roles of our members and their participation in the group project:

|  |  |
| --- | --- |
| **Members** | **Roles** |
| Vanshika | * Create Architecture Diagram. * Deliverable: PNG of the Diagram. * Build VPC, subnets, routing and gateways. |
| Gurkaran | * Build Instances * Deploy web app in public subnet. * Deploy backend database in private subnet. * Deliverable: Step-by-step guide |
| Prarthana | * Automates the solution using CloudFormation. * Deliverable: IaC files + GitHub repo |
| Gulab | * Work on securing the infrastructure. * Deliverable: Report on security challenges. * Finalizes the documentation. |

# Task 1: Architecture Diagram

In the first task, we built an architecture diagram to make our environment clear and well-structured. For that, we used the draw.io tool to create a diagram for our environment which is the blueprint for our project.

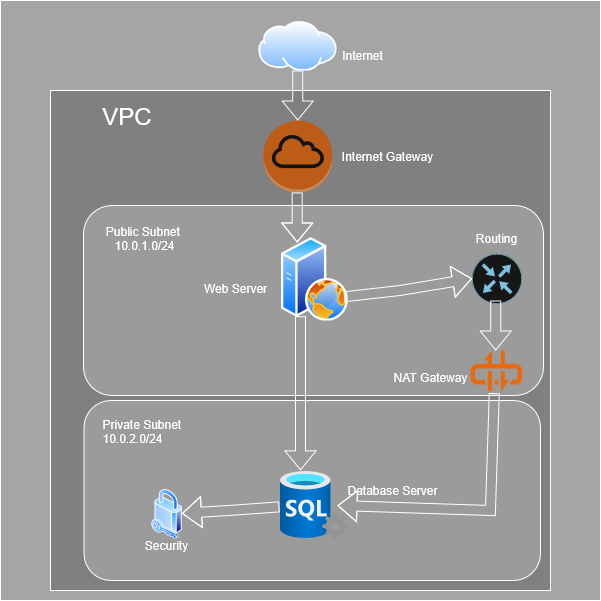


Figure 1 - Shows the architecture diagram for the environment.

# Task 2: Implementation

Following described are the steps we followed to complete the project:

1. Logged in to AWS academy using student login.
2. Selected Learner’s Lab Module.
3. Clicked on Modules>Launch Learners Lab>Start Lab.
4. On the Services, searched VPC.
5. Create a custom VPC:

* Go to VPC>Your VPC>Create VPC.
* Name: gsingh4377-vpc
* IPv4 CIDR: 10.0.0.0/16
* Click Create VPC.

1. Create Subnets:

* Go to Subnets>Create Subnet
* Name-gsingh4377-public-subnet
  + Availability Zone: us-east-1a
  + CIDR:10.0.1/0/24
  + VPC: gsingh4377-vpc
* Name-gsingh4377-private-subnet
  + Availability Zone: us-east-1b
  + CIDR:10.0.2.0/24
  + VPC: gsingh4377-vpc

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2 - Shows the subnets created.

1. Create Internet Gateway:

* Go to Internet Gateways>Create
* Name: gsingh4377-igw
* Attach it to the gsingh4377-vpc

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3 - Shows the internet gateway created.

1. Create Route Tables:

* Name: gsingh4377-private-rt
  + VPC: gsingh4377-vpc
  + Route: 0.0.0.0/0
  + Target- NAT Gateway
* Name: gsingh4377-public-rt
  + VPC: gsingh4377-public-rt
  + Route: 0.0.0.0/0
  + Target- Internet Gateway

A screenshot of a computer

AI-generated content may be incorrect.

Figure 4 - Shows the routes created.

1. Create Security Groups:

* Name: gsingh4377-web-sg
  + Inbound Rules:
    - SSH (22) from 0.0.0.0/0
    - HTTP (80) from 0.0.0.0/0
  + Outbound Rules: Allow all
* Name: gsingh4377-db-sg
  + Inbound Rules: MySQL from gsingh4377-web-sg only
  + Outbound Rules: Allow all

A screenshot of a computer

AI-generated content may be incorrect.

Figure 5 - Shows the security groups created.

1. Launch ec2 instances:

* Go to ec2>Launch instance
* Name: gsingh4377-public-instance
  + AMI: Amazon Linux 2
  + Instance Type: t2.micro
  + Key Pair: Generate a new pair
  + Network: gsingh4377-vpc
  + Subnet: gsingh4377-public-subnet
  + Auto-assign IP: Enable
  + Security Group: gsingh4377-web-sg
  + User data:

#!/bin/bash

yum update -y

yum install httpd -y

systemctl start httpd

systemctl enable httpd

echo "Web Server Running on gsingh4377-public-instance" > /var/www/html/index.html

* Name: gsingh4377-db-server
  + AMI: Amazon Linux 2
  + Instance Type: t2.micro
  + Key Pair: Generate a new pair
  + Network: gsingh4377-vpc
  + Subnet: gsingh4377-private-subnet
  + Auto-assign IP: Disable
  + Security Group: gsingh4377-db-sg
  + User data:
    - Sudo yum update -y
    - Sudo yum install mysql -y

1. Testing

* Access Web Server via Public IP: https://<web-server=ip>

A screenshot of a computer

AI-generated content may be incorrect.

Figure 6 - Shows the web server access to the site.

* Ping to the backend to test connectivity.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 7 - Shows the connectivity test.

# Task 3: Automation

In this task, we will be using CloudFormation to generate the same cloud-based solution.

Step-by-step Implementation:

1. Preparing the template:

* Name: gsingh5377-infra. yaml
* Format: YAML
* Key Components:
  + VPC and subnets.
  + Route tables, security groups, Internet Gateway and NAT Gateway.
  + Ec2 instances for both Web and database.
  + Output: Web Server IP and VPC ID.

1. AWS CloudFormation Console:

* Go to Services> CloudFormation> Create Stack- new resources.
* Upload Template> Choose File> Select gsingh5377-infra.yaml> Click Next.
* Stack Name: group4infrastack>click Next.
* Go with default options on Configure Stack options and acknowledge and create page> Click create Stack.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 8 - Shows the stack created.

1. Verification and Testing:

* Ensure that Web ec2 has public IP and DB ec2 has private IP only.
* Route tables and security groups are configured perfectly.
* Select your stack> Click Outputs> Test the web server ip in browser or via ssh.

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

Figure 9 - Verifies that the server opens in browser.

# Task 4: Security Considerations

In this task, we will discuss some security practices to protect the components of our web application so that the database is completely protected within the private subnet, and only the web application is allowed to the internet.

The three major security considerations and prevention measures are as follows:

1. Backend database being accidentally exposed to the internet.

Prevention Measures:

* The auto-assign IP feature on the database server is disabled while deploying it in a private subnet.
* Internet Gateway only has route to the public subnet.
* Backend has a separate security group for Mysql which uses port 3306 allowing inbound traffic from web server’s security group.

1. Web server facing attacks on ssh/http ports.

* The web server security group allows ssh and http form anywhere for preventing further production and browser access.
* AWS key pair are used for secure SSH login, not the passwords.

1. If the web server is not protected, attackers may breach through the web to the backend.

* The database security group only allows traffic from the Web security group not from entire VPC.
* Outbound rules are monitored and limited to avoid unnecessary excess.

# References

draw.io (n.d.). Untitled Diagram. [https://app.diagrams.net/#](https://app.diagrams.net/)

aws. (n.d.). AWS CloudFormation Documentation. <https://docs.aws.amazon.com/cloudformation/>

awsacademy. (n.d.). AWS Academy Learner Lab [12274]. <https://awsacademy.instructure.com/courses/122774>