Assessment Task
Portfolio of Evidence

ICTCLD505_AT2_PE_TQM_v1.docx

Student Name		**Student Number**	
Unit Code/s & Name/s	ICTCLD505 Implement cloud infrastructure with code		
Cluster Name *If applicable*	N/A		
Assessment Name	Cloud Infrastructure as Code	**Assessment Task No.**	2 of 2
Assessment Due Date		**Date submitted**	/ /
Assessor Name			
Student Declaration: I declare that this assessment is my own work. Any ideas and comments made by other people have been acknowledged as references. I understand that if this statement is found to be false, it will be regarded as misconduct and will be subject to disciplinary action as outlined in the TAFE Queensland Student Rules. I understand that by emailing or submitting this assessment electronically, I agree to this Declaration in lieu of a written signature.			
Student Signature		**Date**	/ /

Instructions to Student	**General Instructions:** You are employed by Uptown IT as a DevOps engineer. You have been assigned to a new project and your task is to create the infrastructure as code (IaC) templates, deploy, test and document the process. Your teacher/assessor will take on the role of the Project Manager assigned to this project by Uptown IT. Read the project documentation provided and familiarise yourself with the Project Scenario or Case Study before proceeding with portfolio tasks. Confirm anything you are not sure about the project with your manager (teacher/assessor). It is essential that you have a clear understanding of the scenario and tasks that you need to complete. This assessment task requires you to complete a project portfolio that is divided into four (4) parts: • PART 1: Prepare to deploy and configure • PART 2: Cloud infrastructure as code (IaC) pre-defined template

- PART 3: Develop and update infrastructure as code (IaC) templates – Create own template
- PART 4: Contingency task and cloud-related knowledge concepts

Materials Required:

- You are required to provide your own storage device. The recommendation for this qualification is an external SSD drive with at least 500 GB capacity, if you need to store a copy of the Virtual Machine (VM). For assessment files only, a 64 GB thumb drive will be sufficient.
- Access to integrated development environment (IDE)
- Access to cloud vendor or 3rd party infrastructure as code service
- information and data sources required to design and implement cloud infrastructure
- Secure shell (SSH) or remote desktop protocol (RDP) client to connect to cloud-hosted instances
- Cloud management console, cloud software development kit or command-line tools.
- Access to the Internet
- Access to Connect (LMS)
- Access to special-purpose tools, equipment and materials to complete the assessment, for example, editing and testing tools.

Online Delivery:

You are to supply your own PC or laptop and peripherals and internet access

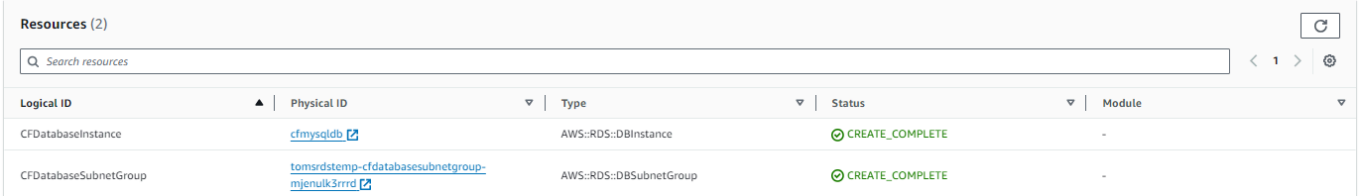
You should have sufficient permissions to be able to install software on your computers e.g. IDE software

You will require access to Microsoft Office or similar

You will require access to a web server or the ability to install a local web server on your computer (instructions for this will be provided)

Documentation:

Uptown IT Scenario or Case Study



Assessment Criteria:

To achieve a satisfactory result, your assessor will be looking for your ability to demonstrate the following key skills/tasks/knowledge to an acceptable industry standard. Demonstrated ability to:

- Identify benefits of infrastructure as code (IaC)
- Determine ways automation leverages cloud platforms
- Evaluate and select infrastructure as code (IaC) services
- Update cloud infrastructure as code
- Work with code templates as required by cloud infrastructure
- Deploy cloud infrastructure as code
- Create and deploy code templates
- Update and delete resources using code templates
- Use cloud infrastructure as code tools as required
- Modify deployed services by modifying templates
- Test and troubleshot templates
- Parameterise and deploy code templates
- Configure resources using the cloud platform console or command line tools
- Create user documentation for cloud infrastructure as code
- Obtain final signoff from required personnel.

Refer to the marking criteria for specific details:

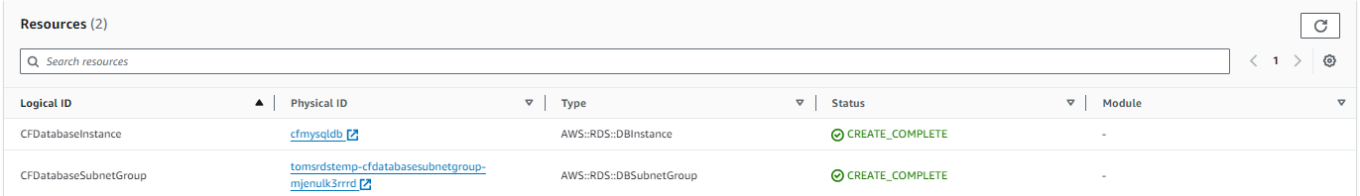
ICTCLD505_AT2_MC_TQM_v1

Details of Location:

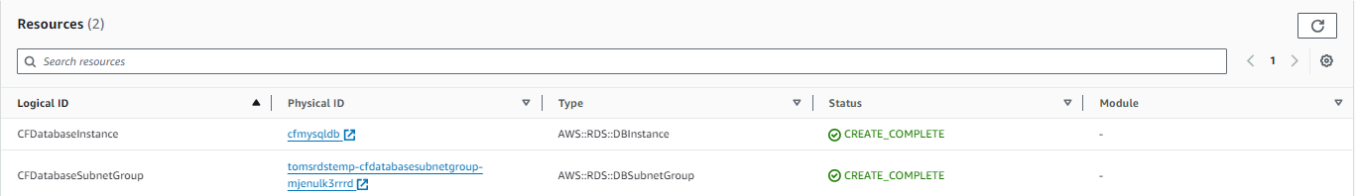
TAFE will provide a simulated work environment in the classroom. Research activities may be conducted in the classroom or at home.

If you are unable to attend a scheduled assessment activity, you must notify your teacher before the assessment is due and supply a doctor's certificate and approval from the team manager for an extension.

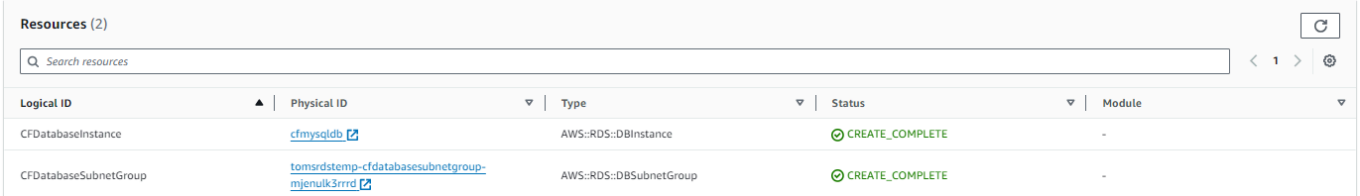
Time Restrictions:



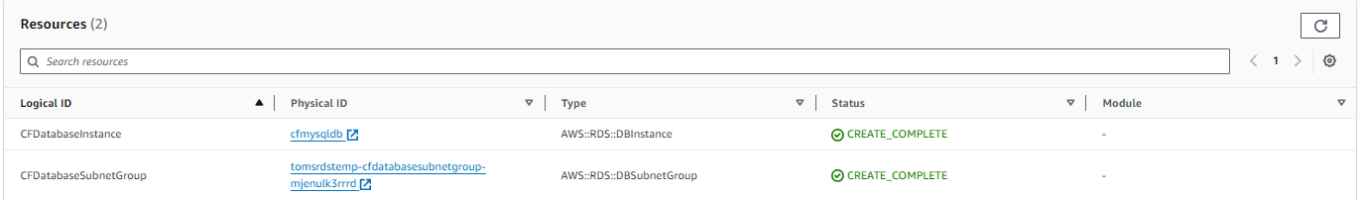
	This assignment is designed to take place over 8 weeks or approximately 32 hours. The student is expected to attend classes as per timetable details and should be able to commit up to 3 hours per week of their own time to study or study related activities. **Interactions:** Teamwork skills are essential in the IT industry therefore you should work in teams to consult and collaborate on practical activities. However, each student must complete the assessment tasks individually (unless indicated).



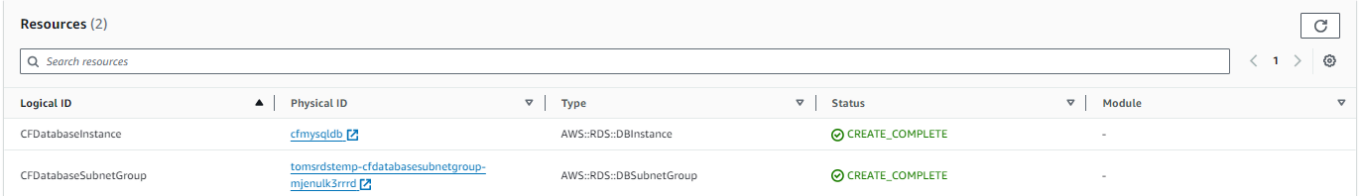
	Level of Assistance Permitted: Staff cannot directly show students answers or solutions but support and guide them to complete tasks individually. Teachers and tutors should be available in class, and accessible by email for students working from home. **Reasonable Adjustments:** Reasonable adjustments are available to students for a variety of reasons, including: disability, language, literacy and numeracy (LLN) problems or extenuating circumstances. Talk to your teacher, counsellor, or disability officer if you require extra support or an extension based on the conditions identified. **Number of Attempts:** You will receive up to two (2) attempts at this assessment task. Should your 1st attempt be unsatisfactory (U), your teacher will provide feedback and discuss the relevant sections / questions with you and will arrange a due date for the submission of your 2nd attempt. If your 2nd submission is unsatisfactory (U), or you fail to submit a 2nd attempt, you will receive an overall unsatisfactory result for this assessment task. Only one re-assessment attempt may be granted for each assessment task. **For more information, refer to the Student Rules.** **Work, Health and Safety:** The work environment should be assessed for safety prior to class. Special consideration should be taken regarding potential ICT related hazards such as tripping hazards, electromagnetic radiation, ergonomics, and posture. TAFE Queensland health and safety policies and procedures should be followed at all times.
Submission details (if relevant)	**Evidence Required to be Submitted:** Insert your details on the cover page and sign the Student Declaration. Include this template with your submission. **Submission via Connect:** Upload a single file into Assessment 2 (AT2) Assignment Folder in Connect. Multiple files can be compressed into a single file.



	Name the file: ICTCLD505_AT2_Surname_Student Number



	TAFE Queensland Learning Management System: Connect url: https://connect.tafeqld.edu.au/d2l/login ● Username; 9 digit student number For Password: Reset password go to: https://passwordreset.tafeqld.edu.au/default.aspx
Instructions to Assessor	**Online Delivery:** Please revise and modify the Instructions to Student section if you are delivering online. **Specifications of Assessment:** To be judged competent in this assessment item the student is required to demonstrate competence in all indicators shown in the marking guide. Gather evidence to demonstrate consistent performance in conditions that are safe and replicate the workplace. Noise levels, production flow, interruptions and time variances must be typical of those experienced in the web development field of work and include access to: ● cloud development environment ● project requirements Ensure that students read and familiarise themselves with the Project Scenario and the Client provided relevant files and/or resources before attempting the assessment. **Storage Devices:** Students are required to provide their own storage device. The recommendation for this qualification is an external SSD drive with at least 500 GB capacity, if you need to store a copy of the Virtual Machine (VM). For assessment files only, a 64 GB thumb drive will be sufficient. **Assessor to Provide:** ● Students are required to provide their own storage device. The recommendation for this qualification is an external SSD drive with at least 500 GB capacity, if you need to store a copy of the Virtual Machine (VM). For assessment files only, a 64 GB thumb drive will be sufficient.



- Access to integrated development environment (IDE)
- Access to cloud vendor or 3rd party infrastructure as code service
- information and data sources required to design and implement cloud infrastructure
- Secure shell (SSH) or remote desktop protocol (RDP) client to connect to cloud-hosted instances
- Cloud management console, cloud software development kit or command-line tools.
- Access to the Internet
- Access to Connect (LMS)
- Access to special-purpose tools, equipment and materials to complete the assessment, for example, editing and testing tools.

Online Delivery:

- Students to supply their own PC or laptop and peripherals and internet access
- Students should have sufficient permissions to be able to install software on their computers e.g. IDE software
- Students will require access to Microsoft Office or similar
- Students will require access to a web server or the ability to install a local web server on their computer (instructions for this will be provided)

Documentation:

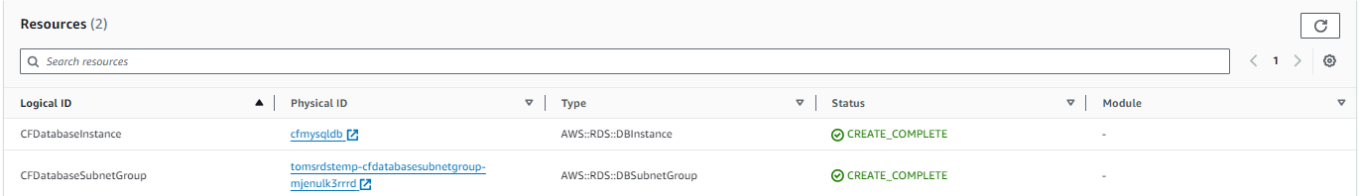
Uptown IT Scenario or Case Study

Level of Assistance Permitted:

Teachers and tutors should be available in class, and accessible by email for students working from home. Staff cannot directly show students answers but support and guide them to complete tasks individually. Students with disability will receive reasonable adjustments.

Interactions:

Teamwork skills are essential in the IT industry therefore you should work in teams to consult and collaborate on practical activities. However, each



	student must complete the assessment tasks individually (unless indicated). **Contingencies:** Reasonable adjustment is available to students for a variety of reasons, including: disability, language, literacy and numeracy (LLN) problems or extenuating circumstances. **Work, Health and Safety:** The work environment should be assessed for safety prior to class. Special consideration should be taken regarding potential ICT related hazards such as tripping hazards, electromagnetic radiation, ergonomics, and posture. TAFE Queensland health and safety policies and procedures should be followed at all times.
Note to Student	An overview of all Assessment Tasks relevant to this unit is located in the Unit Study Guide.

Project Scenario

ROLE

You are employed by Uptown IT as a DevOps engineer. You have been asked by your manager to transition an application under development to a staging platform. The application developed is a LAMP stack that must be separated into microservices. The configurations made in this process will eventually be used in production with minor modifications. This activity assures management that the necessary preparations have been made in lieu of handover to the operations team.

Your teacher/assessor will take on the role of the Project Manager assigned to this project by Uptown IT.

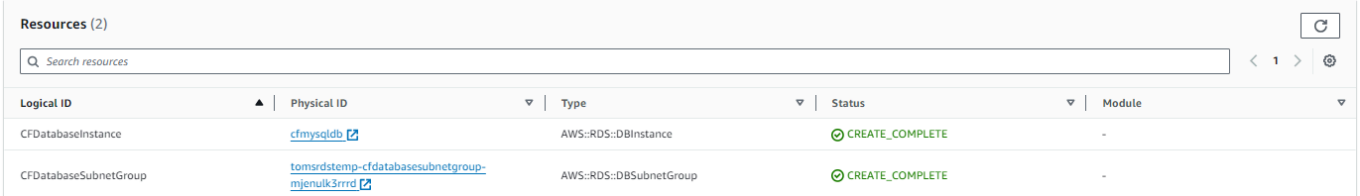
FILES

Project files are contained in the attached folder.

This is a PHP application that uses a MySQL database to store data persistently. There is also a file upload feature that needs the webserver write permission to a portion of the filesystem.

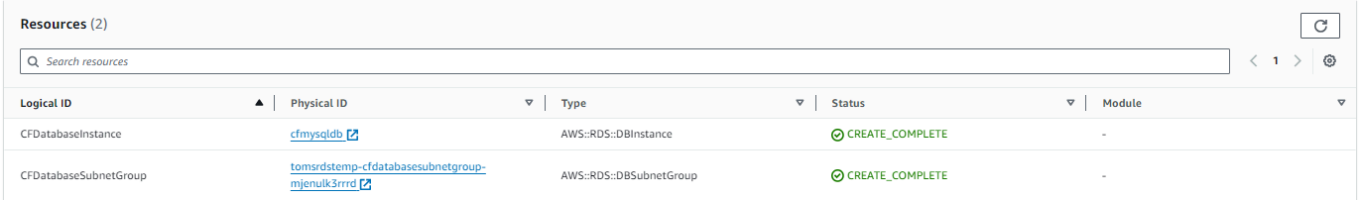
The download includes:

- Website files and folders



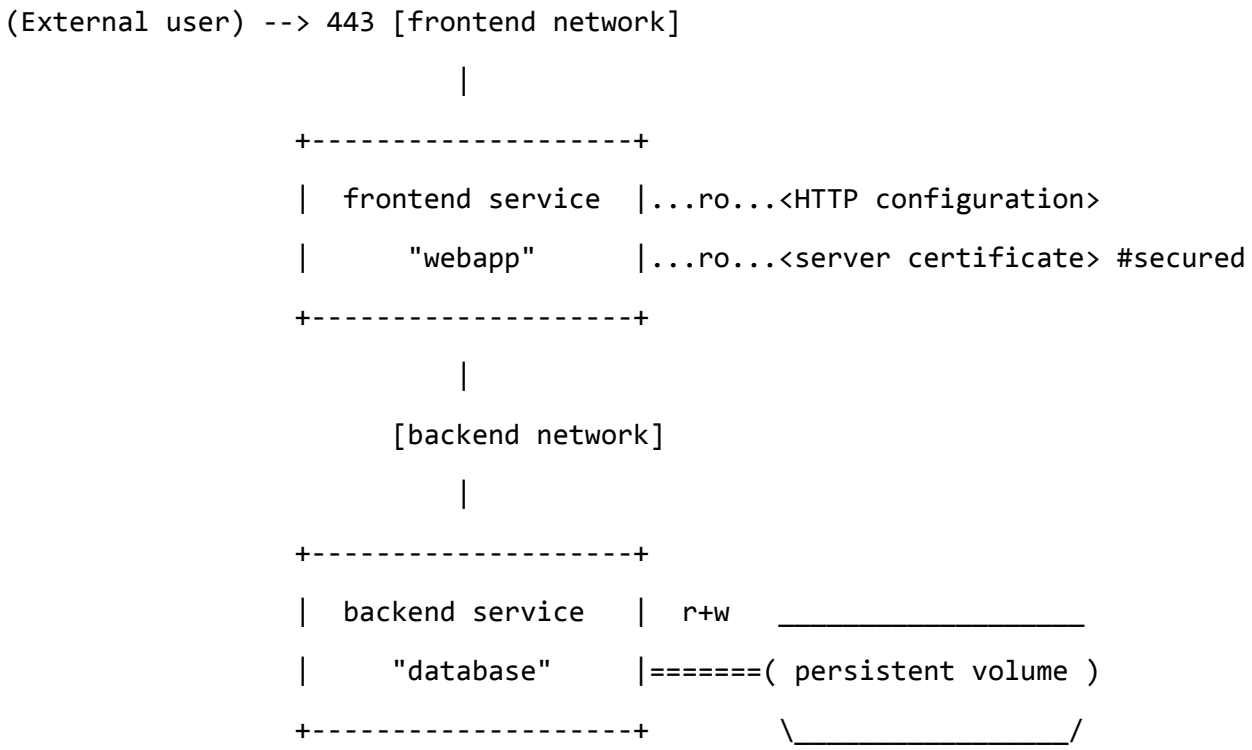
- Controller folder
- Model folder
- View folder
- Index.php

- A README file
- The SQL file (gorgeous_cupcakes_v1.sql)



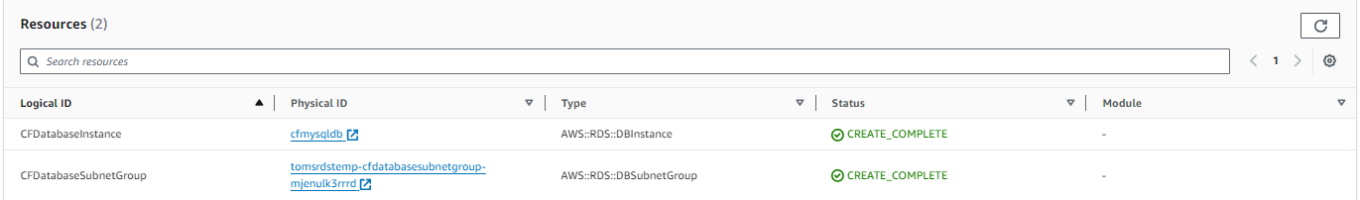
PROJECT

The project consists of a web application and a database. The services communicate with each other on an isolated back-tier network, while the frontend is also connected to a front-tier network and exposes port 443 for external usage.



The application is composed of the following parts:

- 2 services: web application and database
- 1 secret (HTTPS certificate), injected into the frontend
- 1 configuration (HTTP), injected into the frontend
- 1 persistent volume, attached to the backend
- 2 networks



Answers to this assessment portfolio (Parts 1 to 4) are to be word-processed in a separate document. Name the document:

<studentsurname>_ICTCLD505_portfolio

PART 1 Prepare to deploy and configure

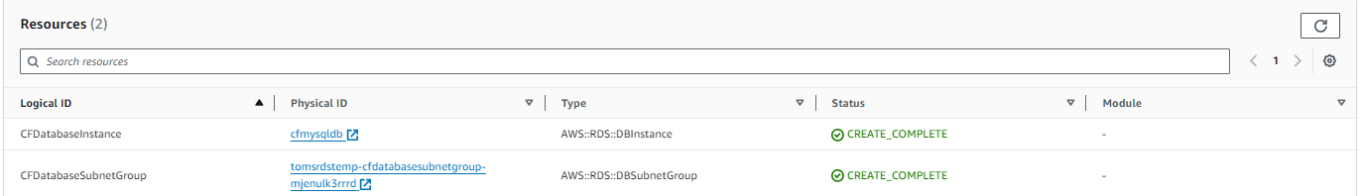
In this research stage of the project, you need to explore and evaluate the technology and tools most suited to the work required. The following questions provide some guidance for your research.

Answer each question in sufficient detail to allow the assessor to follow your decision-making process.

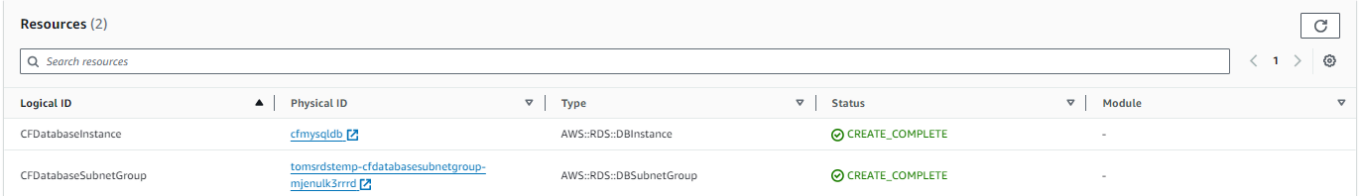
1.1 Review the scenario presented. Research, evaluate, and select an appropriate cloud platform for the portfolio project. Provide a detailed justification for your selection.

Referring to Implement cloud infrastructure with code guide (page 9 Learner Guide), the most appropriate cloud platform for transitioning a LAMP stack application to a microservices architecture is Amazon Web Services (AWS). AWS is noted for its global operations, extensive services, and flexible pricing plans, making it an ideal platform for such a transition. It offers specific services for data storage (AWS Athena, Parquet files) , media streaming (https://aws.amazon.com/solutions/media-entertainment/streaming-media/) , and quantum computing (Amazon Braket https://aws.amazon.com/braket/), which could be useful and future proof the security of UpTown IT .

Compare and contrast three (3) infrastructure as code (IaC) technologies that could be utilised for the scenario presented and are compatible with the cloud platform selected in 1.1 above.



Feature/Aspect	AWS CloudFormation	Terraform	Ansible
Integration	Pros: Deeply integrated with AWS.Cons: Limited to AWS, not ideal for multi-cloud scenarios.	Pros: Supports multiple cloud platforms, including AWS.Cons: Integration quality varies between providers.	Pros: Wide integration capability with cloud services.Cons: May require additional modules for full AWS integration.
Configuration Language	Pros: Uses widely known JSON/YAML.Cons: Requires AWS-specific template knowledge.	Pros: HCL allows for expressive configurations.Cons: HCL is unique to Terraform, requires specific learning.	Pros: YAML is simple and widely used.Cons: YAML can be prone to errors due to indentation.
Service Coverage	Pros: Extensive AWS service support.Cons: Exclusively for AWS services.	Pros: Broad service support across various technologies.Cons: Dependent on providers for service updates.	Pros: Supports a wide range of services via modules.Cons: Coverage dependent on community modules.
State Management	Pros: Effective state management for AWS resources.Cons: Limited to AWS resource states.	Pros: Robust state tracking for multiple services.Cons: Complexity increases with state management at scale.	Pros: Manages desired state effectively.Cons: Less comprehensive compared to Terraform and CloudFormation.
Security and Compliance	Pros: Aligns well with AWS security and compliance.Cons: Primarily focused on AWS standards.	Pros: Adapts to the security standards of multiple clouds.Cons: Relies on external configurations for full compliance.	Pros: Good for standard security automation.Cons: Requires additional tools for advanced compliance needs.
Ease of Learning	Pros: Integrates well within AWS ecosystem.Cons: Steep learning curve for AWS specifics.	Pros: Structured and logical language.Cons: HCL syntax may be unfamiliar to new users.	Pros: User-friendly for beginners with YAML.Cons: Some Ansible specifics can be challenging to master.
Agent Requirement	Pros: N/A (AWS service).Cons: N/A (AWS service).	Pros: Agentless, operates as a CLI tool.Cons: Requires	Pros: Agentless, reduces system overhead.Cons:



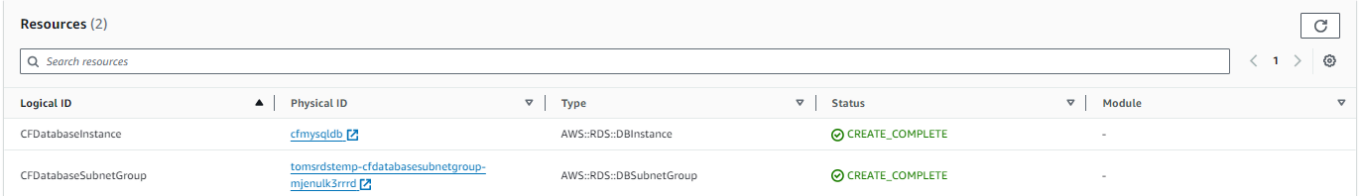
		setup and understanding of CLI operations.	Sometimes limited in controlling remote systems.
Error Handling	Pros: Clear error messages within AWS context.Cons: Sometimes lacks detailed error explanations.	Pros: Detailed error reporting.Cons: Error messages can be cryptic and complex.	Pros: Generally user-friendly error messages.Cons: Error clarity can vary based on the module used.
Flexibility	Pros: Highly optimized for AWS environments.Cons: Not flexible for non-AWS resources.	Pros: Highly flexible and adaptable to various environments.Cons: Can be complex in heterogeneous environments.	Pros: Flexible in diverse IT environments.Cons: Sometimes less efficient for complex orchestration tasks.
Community Support	Pros: Strong support from AWS community.Cons: Focused only on AWS-related issues.	Pros: Large and active community.Cons: Community support varies by provider.	Pros: Extensive community support and documentation.Cons: Quality of community contributions can vary.

~~Researching Scenario below:~~

~~Sources:~~
~~https://en.wikipedia.org/wiki/Systems_development_life_cycle~~

~~https://en.wikipedia.org/wiki/DevOps~~
~~https://www.redhat.com/en/topics/devops/devops-engineer~~

~~Situation. IT consultant as a DevOps engineer.~~
~~What is DevOps~~
~~DevOps is shortened for software development (Dev) and IT Operations (Ops), it is a software development methodology which aims to improve and shorten a system's development life cycle, meaning getting ideas into actions in a technologically practical sense.~~
~~So basically they are like the 'plumbers' of the IT world, trying to integrate software and applications together to work and run in closed test environments for fixes or from the other side launching specific applications from the ground up.~~
~~Applying generalised IT knowledge and funnel it into specific business and needs and produce~~



~~results is what I'd consider a Devops Engineer to be from a purely business perspective.~~

~~Lamp Stack?~~

~~Lamp stack is a bundle of four different software technologies that developers use to build websites and web applications. LAMP is an acronym for the operating system, Linux; the web server, Apache; the database server, MySQL; and the programming language, PHP. (https://aws.amazon.com/what-is/lamp-stack)~~

~~Linux: Derived from the principles of Unix. Providing extreme modularity in configuration.~~
~~Apache: is a webserver application that is able to handle HTTP requests and provide web content to web browser clients. Apache typically addresses request handling by creating a new thread or process for each connection. This allows for an isolated approach between requests however asynchronous code would be multitudes more efficient, such as how nginx fundamentally handles requests.~~
~~MySQL: MySQL is database manager developed my oracle that is based on SQL.~~

~~SQL is the database language that MySQL uses to interact with the database itself. RDBMS's use primary keys, foreign keys, rows and tables to store data.~~

~~PHP: While Apache handles the actual connection requests of specific users. PHP gets requests from Apache if they include PHP in them, and can create dynamic content on a website, it can process user inputs and turn them into query strings like URL params (.php/?tomsexample=360), cookies and can interact with MySQL using PHP Data Objects (PDOs). basic examples include get and post requests.~~

For each technology or tool explain their features and if they rely on a domain-specific language or use standard notation language (e.g., yaml or json).

AWS CloudFormation

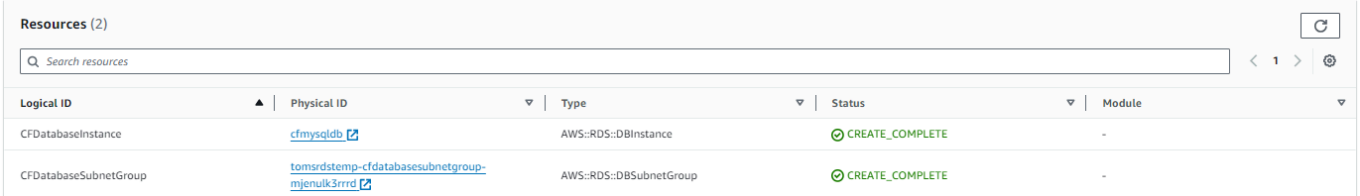
Features:

Allows users to create and manage a collection of related AWS resources, provisioning and updating them in an orderly and predictable fashion.

Language:

Uses JSON or YAML for its templates. These are standard notation languages that are widely used for data serialization.

Key Aspects:



Integrated with AWS, offering seamless deployment of AWS resources.

Supports automated dependency management and rollback features.

Template-based management enables replication and version control of infrastructure setups.

Terraform

Features:

An open-source IaC tool that allows for the building, changing, and versioning of infrastructure safely and efficiently.

Language:

Utilizes HashiCorp Configuration Language (HCL), which is a domain-specific language. HCL is designed to be both human-readable and machine-friendly.

Key Aspects:

Platform-agnostic, capable of managing multi-cloud infrastructures.

Maintains state files to track resource deployment.

Supports modular design, enhancing reusability and maintainability of infrastructure code.

Ansible

Features:

An open-source automation platform used for IT tasks such as configuration management, application deployment, intra-service orchestration, and provisioning.

Language:

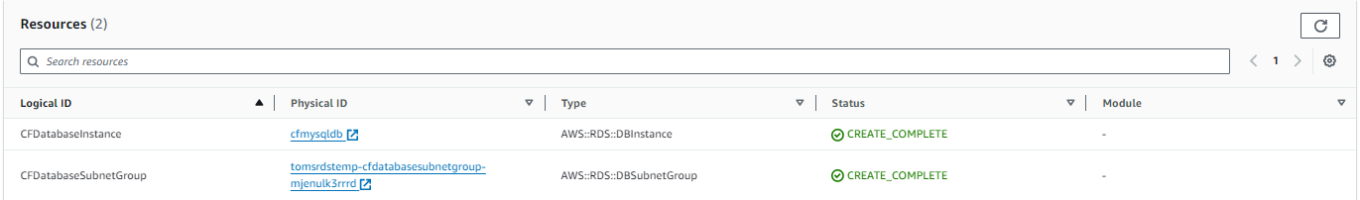
Employs YAML for its playbooks, a standard data serialization language.

Key Aspects:

Agentless architecture, simplifying operations and reducing system overhead.

Emphasizes simplicity and ease-of-use, with a focus on automation.

Robust community support and integration with a variety of cloud and IT environments.



Evaluate and select the most appropriate IaC technology or tool for the scenario presented. Provide a detailed justification of your selection.

Based on the prior explanation of the tools and features this is a summary that compares and weighs in each one that would be the most ideal for UpTown IT.

AWS CloudFormation

Suitability: Excellent for AWS-specific environments. Its deep integration simplifies the management of AWS resources.

Considerations: Limited to AWS; less flexibility for non-AWS resources.

Terraform

Suitability: Good for multi-cloud strategies. It provides a high degree of flexibility and state management capabilities.

Considerations: The need to manage state files and learn HCL.

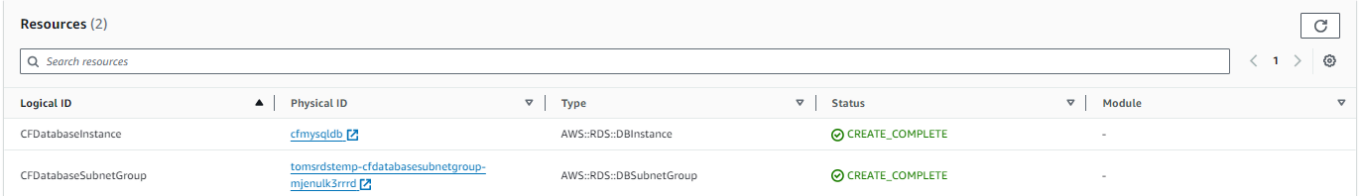
Ansible

Suitability: Great for automation and configuration management. Its agentless architecture simplifies operations.

Considerations: Less direct support for AWS microservices tools and more suited to configuration management than infrastructure provisioning.

Selection and Justification

Selecting AWS CloudFormation for the portfolio project to transition a LAMP stack to a microservices architecture on AWS is justified due to its specific alignment with the AWS ecosystem. CloudFormation's native integration with AWS ensures optimal management and deployment of AWS-specific resources, which is critical for a project relying heavily on AWS services like EC2, RDS, and S3. The tool's template-based configuration approach is ideal for handling the complexity associated with microservices. It simplifies replicating environments, managing intricate architectures, and automates dependency management,



which are key factors in a microservices setup. Additionally, CloudFormation's features like rollback and error handling are essential for maintaining stability and consistency during deployments, crucial for the project's success.

In comparison, while Terraform offers broader flexibility and Ansible excels in configuration management, neither provides the same depth of integration or the specialized features for AWS resource management as CloudFormation. This makes CloudFormation the most practical and efficient choice for this AWS-centric project.

1.2 Research and review at least three (3) benefits of using infrastructure as code (IaC) for the scenario presented. Explain each benefit in detail and provide examples to illustrate your answer.

1. Enhanced Disaster Recovery and Risk Management:

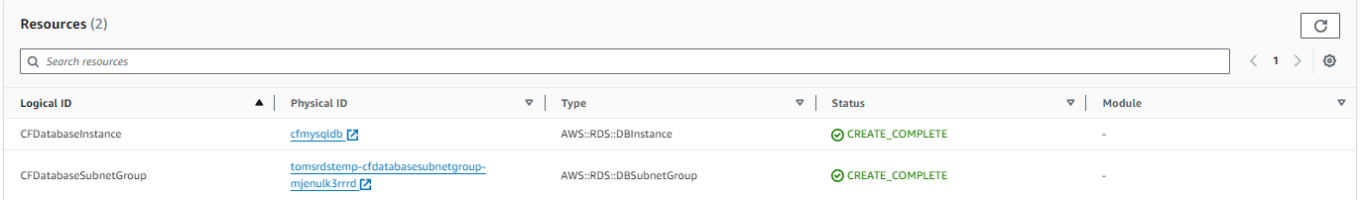
Detail: IaC improves disaster recovery by enabling rapid and accurate rebuilding of infrastructure. By codifying infrastructure, teams can quickly redeploy their entire infrastructure from the IaC templates, ensuring minimal downtime.
Example: If an AWS server hosting a microservice fails, IaC can automatically redeploy the service on a new server with identical configurations, ensuring continuous service availability and minimizing the impact on end users.

2. Scalability and Flexibility in Resource Management:

Detail: IaC allows for easy scaling of infrastructure resources to match demand. It enables dynamic adjustment of resources based on workload, enhancing the application's responsiveness and efficiency.
Example: During a surge in user traffic, IaC can automatically scale up the database and server resources for your microservices, ensuring smooth performance. When the traffic subsides, it scales down the resources, optimizing cost efficiency.

3. Streamlined Compliance and Standardization:

Detail: IaC standardizes infrastructure setups, ensuring compliance with organizational and regulatory standards. It acts as a blueprint, ensuring that every deployment adheres to predefined configurations.
Example: In deploying a new microservice, IaC ensures it complies with your organization's security protocols (like specific firewall settings or encryption standards). This uniformity simplifies compliance audits and enhances overall security.



1.3 Research and determine at least three (3) ways or techniques in which automation can leverage cloud platforms. For each automation technique, provide a detailed explanation and example (s).

Configuration Management: It ensures that all cloud resources follow predefined configurations, reducing inconsistencies and manual oversight. For instance, tools like Ansible or Terraform can automate the setup of cloud environments, ensuring that every deployment adheres to company standards.(https://www.otava.com/blog/how-to-leverage-cloud-automation-to-boost-productivity-and-reduce-strain-on-skilled-labor/)

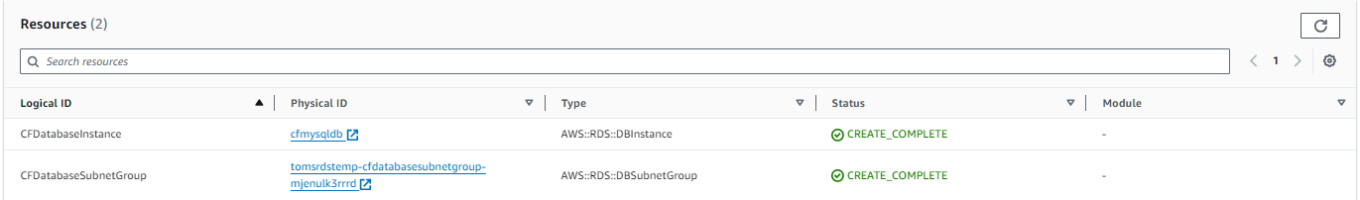
Second-Generation Cloud Automation:

Advanced cloud automation, or second-gen automation, goes beyond simple task automation to encompass infrastructure and services as code. This involves utilizing cloud-native services and applications, and integrating complex workflows that leverage AI and ML, to enable smarter data usage. For instance, automating data flow management between services, such as between an ERP system and a CRM, can optimize business intelligence and operations. (https://technative.io/what-is-advanced-cloud-automation/)
Digital Process Automation (DPA):

Platforms like Microsoft's Power Automate represent the accessible edge of second-gen cloud automation. They connect a multitude of business systems and utilize Azure Cognitive Services to manage daily business workflows intelligently. However, integrating DPA with legacy systems may require additional expertise, underscoring the importance of adapting automation strategies to the existing IT landscape.(https://powerautomate.microsoft.com/en-au/digital-process-automation/)

1.4 Identify three (3) types of problems or errors that can occur when infrastructure is implemented as code as opposed to manual processes. In your response consider testing and debugging techniques.

Vendor Lock-in: Selecting a cloud-specific IaC tool can lead to vendor lock-in. For instance, AWS CloudFormation is not compatible with other cloud providers, which can become an issue if a multi-cloud strategy is adopted. To address this, cloud engineers often use cloud-agnostic IaC tools like Terraform or Pulumi.



Security Misconfigurations: IaC can inadvertently propagate misconfigurations across an entire infrastructure if not carefully managed. Security policies as code should be enforced using tools like Open Policy Agent to prevent deployments with potential security flaws.

Code Redundancy and Drift: IaC can suffer from code duplication, leading to drift and inconsistencies. The principle of "Don't Repeat Yourself" (DRY) must be adhered to, often through the use of IaC frameworks that support modular and reusable code structures like Terraform modules or Terragrunt.

To mitigate these issues, common testing and debugging techniques include:

Unit Testing: Using frameworks like ChefSpec for Chef or Pytest for Python scripts to test individual components of the code.

Integration Testing: Tools like Test Kitchen or InSpec can be used to simulate the deployment of code in an isolated environment to check for interactions between components.

Policy Testing: Implementing policy as code testing, ensuring that security and compliance policies are applied correctly before deployment.

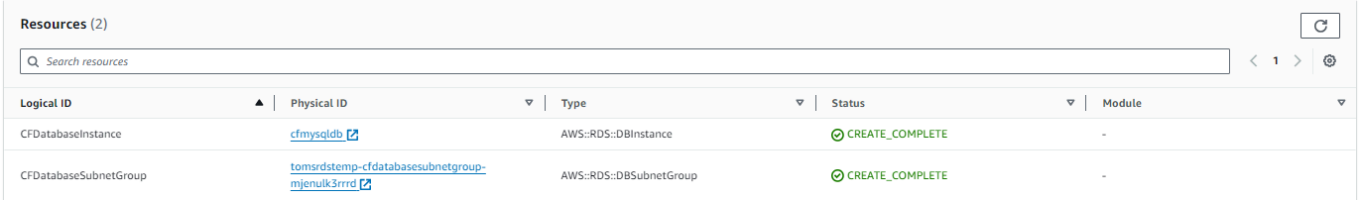
Linter Tools: Tools like Terraform Lint or CloudFormation Linter can analyze code to ensure it adheres to best practices and catch errors before deployment.

1.5 For each problem or error identified, assess the potential consequences. Provide examples to illustrate your answer in reference to the scenario presented.

Vendor Lock-in Consequences: If an organization's IaC scripts are written for a specific cloud provider, they may face significant challenges if they decide to migrate to another cloud or adopt a multi-cloud strategy. This could result in having to rewrite IaC configurations from scratch, leading to increased costs and delayed migration projects.

Security Misconfiguration Consequences: If IaC templates contain security misconfigurations, they could be replicated across the entire infrastructure, leading to widespread vulnerabilities. This could result in data breaches or compliance failures. For example, an improperly configured security group in AWS might expose sensitive databases to the public internet.

Code Redundancy and Drift Consequences: Duplicated and hard-coded IaC configurations can lead to environment drift where production, development, and testing environments become inconsistent with each other. This may result in deployment issues, service disruptions, and increased maintenance overhead. For instance, an update applied to one environment but omitted in another due to code redundancy might lead to unexpected behavior or system outages.



1.6 Review the functionality of the scenario presented and investigate any required industry or organisational standards, and legislative requirements necessary for the project. Document your findings.

Sector-Specific Legislation (e.g., My Health Records Act 2012, Corporations Act 2001):

Example: For a healthcare app, IaC scripts include safeguards such as conducting regular backups and enabling strict access controls to health records to adhere to the My Health Records Act. In the financial sector, IaC ensures that trading systems log all transactions with the necessary details to satisfy record-keeping demands of the Corporations Act. My dad worked on this and helped release MHR to the public. My dad created the backend release for myhealthrecord, he was situated right next to the policy makers inside queensland health and there would always be rigorous rules to follow.
Examples for Industry Standards:

Australian/New Zealand Standards (AS/NZS ISO/IEC 27001):

Example: An IaC script is designed to automatically set up and configure logging and monitoring on all new cloud server instances to adhere to information security management practices. The script ensures that all logs are retained according to the defined retention policy and that access to these logs is restricted and monitored.

ASD Cybersecurity 'Essential Eight':

Example: IaC configurations enforce application whitelisting policies by scripting firewall rules and access control lists that only allow approved software to run on cloud-based virtual machines, in line with the ASD’s mitigation strategies.

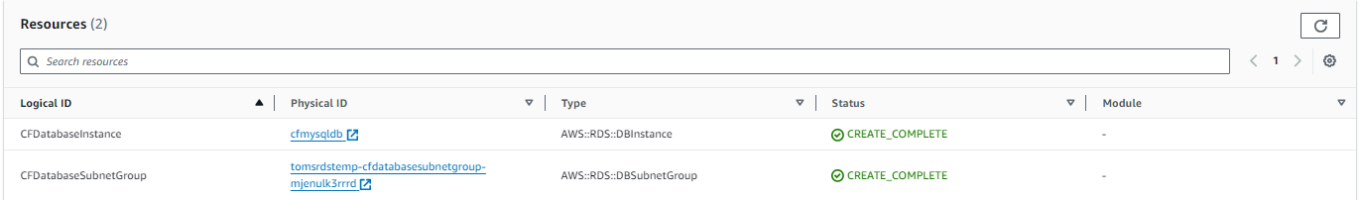
ISO/IEC 27017 and ISO/IEC 27018:

Example: IaC scripts specify encryption for data at rest and in transit settings for cloud storage resources, affirming the organization's commitment to protecting personal and sensitive information as per ISO/IEC 27018 privacy guidelines.

Examples for Organisational Standards:

Custom Compliance Controls for Financial Institutions:

Example: For an online banking platform, IaC automation includes setting up multi-factor authentication (MFA) for all cloud services and privileged accounts, as specified in the organization's internal



cybersecurity framework which reflects the financial industry's stringent requirements.

DevOps Practices Incorporating CI/CD:

Example: Integrating IaC with a CI/CD pipeline where automated tests are executed before any infrastructure changes are deployed. Version control systems tag each IaC commit, ensuring robust change management and tracking.

Examples for Legislative Requirements:

Privacy Act 1988 (Cth) and APPs:

Example: IaC scripts are written to deploy databases with configurations that anonymize private user data automatically, meeting APPs by introducing data-minimization techniques.

Information Security Manual (ISM):

Example: Utilizing IaC to enforce ISM guidelines, an automated procedure is established to patch operating systems and applications within a brief time window after new patches are released.

Notifiable Data Breaches (NDB) Scheme:

Example: IaC includes an automated process for flagging potential data breaches by monitoring for unusual data access patterns, enabling swift notification and response as required by the NDB scheme.

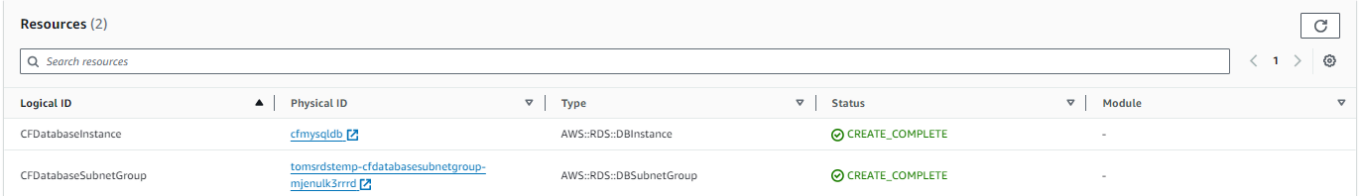
Examples for Industry Compliance Frameworks:

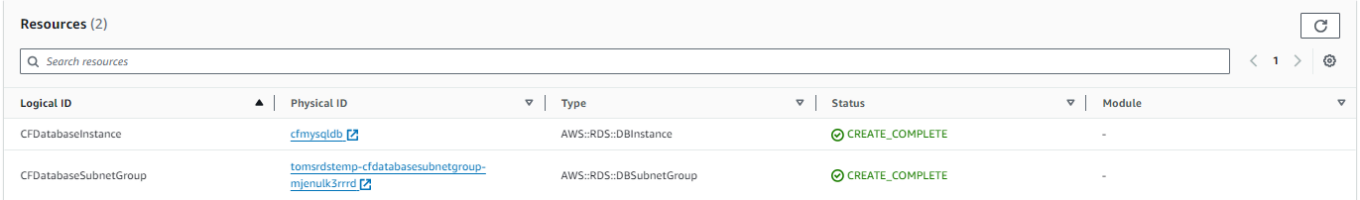
APRA Prudential Standard CPS 234:

Example: IaC deployment scripts for financial systems are coded to ensure all sensitive data storage systems have encryption at both rest and in transit, as well as deploying tools for real-time threat detection and response capabilities.

ACSC Guidelines:

Example: Automatic configuration of network intrusion detection systems using IaC to comply with ACSC guidelines, ensuring continuous surveillance and security of cloud infrastructure.




Resources (2)
Search resources
1
Logical ID
Physical ID
Type
Status
Module
CFDatabaseInstance
cfmysqldb
AWS::RDS::DBInstance
CREATE_COMPLETE
-
CFDatabaseSubnetGroup
tomsrdstemp-cfdatabasesubnetgroup-mjenulk3rrrd
AWS::RDS::DBSubnetGroup
CREATE_COMPLETE
-

PART 2 - Cloud infrastructure as code (IaC) pre-defined template

2.1 Review the pre-defined template provided and determine and explain the resources and dependencies used. Provide detailed explanations for each resource.

VPC (Virtual Private Cloud):

Details: Constructs a secure, isolated virtual network within AWS. It's configured with a specific IP range (CFVpcCIDR) and includes an Internet Gateway (CFInternetGateway) for external connectivity. The VPC is segmented into both public (CFPublicSubnet) and private subnets (CFPrivateSubnet and CFPrivateSubnet2), offering a layered network structure for different resource needs.

Functionality: Ensures secure and controlled network environment for the application, segregating internal and external traffic effectively.

Security Groups:

Details: These are virtual firewalls defined within the VPC. The CFWebServerSG allows HTTP and SSH traffic, CFHTTPSSG is configured for HTTPS traffic, and CFDataBaseSG permits MySQL connections. Each group is tailored to the specific security requirements of different components (web server, database).

Functionality: Provides a robust security mechanism to regulate access and protect resources from unauthorized or harmful traffic.

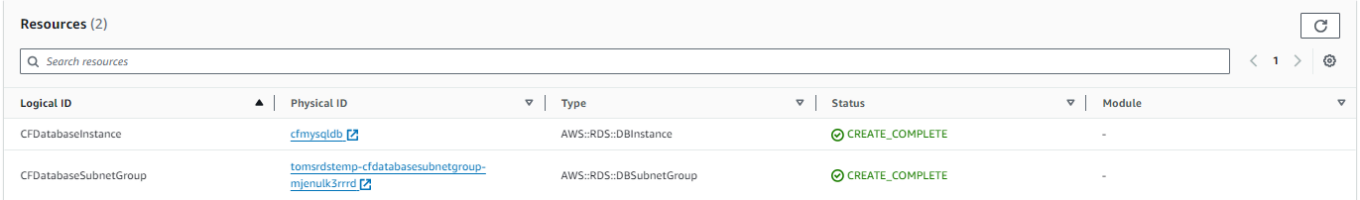
EC2 Instance:

Details: An AWS EC2 instance (CFVPCEC2) is specified, linked to the public subnet for external accessibility and attached to security groups for web and HTTPS traffic. UserData scripts are included for automatic installation and configuration of the LAMP stack, along with application deployment.

Functionality: Acts as the primary compute resource, hosting and running the web application, with automated setup for ease of deployment.

RDS Instance:

Details: Sets up a managed MySQL database (CFDatabaseInstance) in the VPC's private subnets, utilizing the database security group for secure network access. Configurations



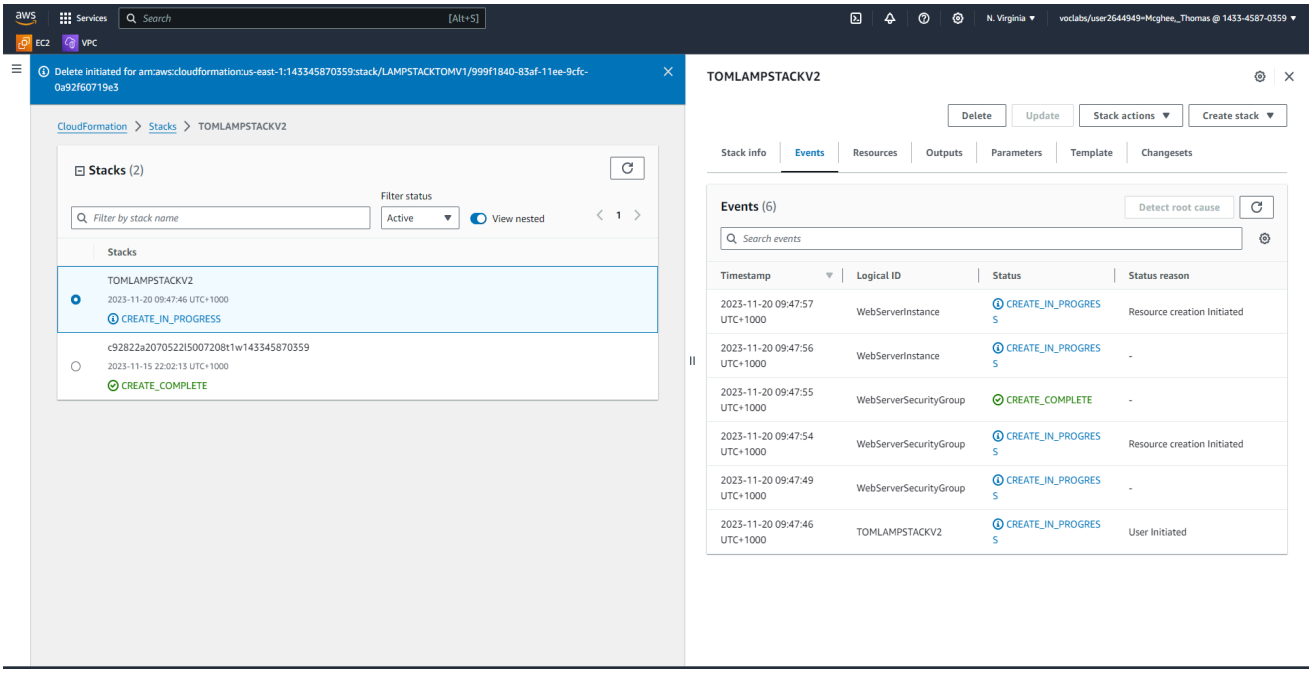
include database specifics like name, user credentials, storage capacity, and multi-AZ deployment option for higher availability.

Functionality: Provides a reliable, scalable, and managed database service, essential for data persistence in the application, with enhanced availability and security.

2.2 Demonstrate knowledge of the selected IaC syntax by using the CLI (Command-line interface) or console to configure resources and confirm/orchestrate the deployment. Perform the following tasks:

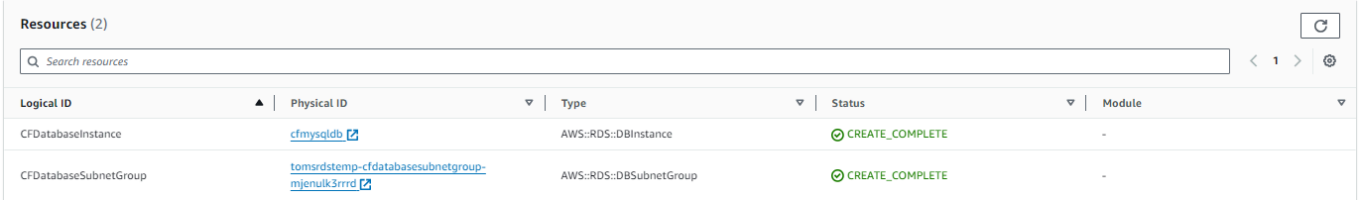
a) Start service

Starting cloudformation services



b) Configure template

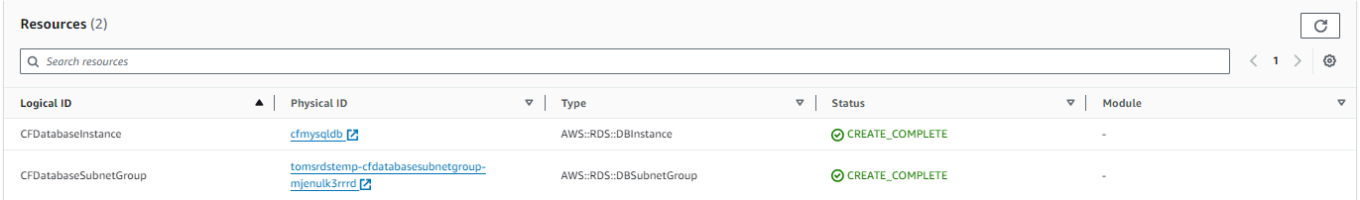
Put resource part of template here (YAML template)

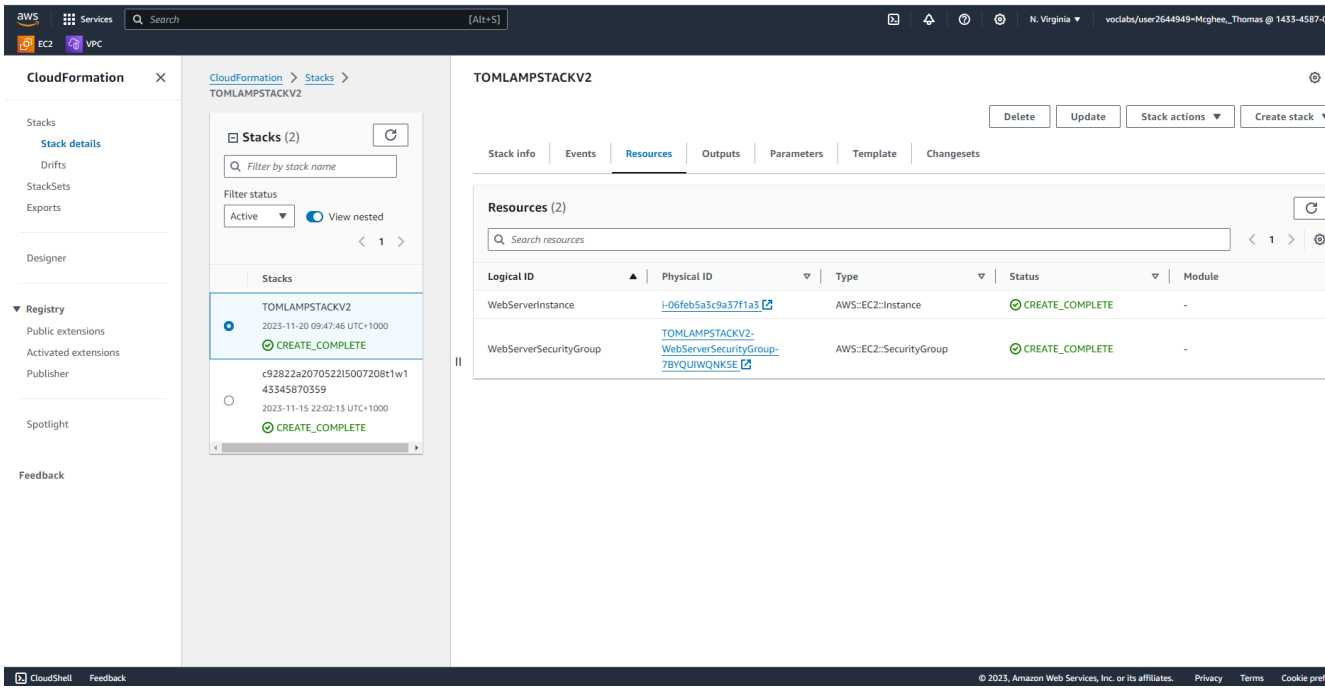


```
Resources:
  WebServerInstance:
    Type: 'AWS::EC2::Instance'
    Metadata:
      'AWS::CloudFormation::Init':
        configSets:
          InstallAndRun:
            - Install
            - Configure
        Install:
          packages:
            yum:
              mysql: []
              mysql-server: []
              mysql-libs: []
              httpd: []
              php: []
              php-mysql: []
          files:
            /var/www/html/index.php:
              content: !Join
                - ''
                - - |
                    <html>
                  - |2
                    <head>
                  - |2
                    <title>AWS CloudFormation PHP Sample</title>
                  - |2
                    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
                  - |2
                    </head>
                  - |2
                    <body>
                  - |2
                    <h1>Welcome to the AWS CloudFormation PHP Sample</h1>
                  - |2
                    <p/>
                  - |2
                    <?php
                  - |2
                      // Print out the current data and time
                  - |2
                      print "The Current Date and Time is: <br/>";
```

c) Deploy resources

Take screenshot of successful resource deployment
“Run template and take screenshots of resources that you’ve deployed like ec2 instances and whatever resources like databases and put in here”

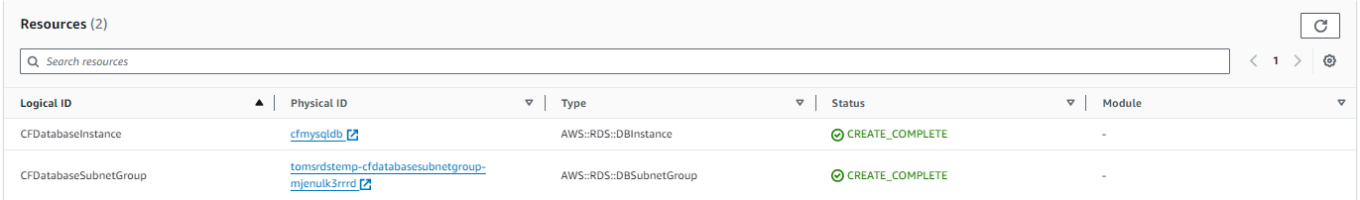




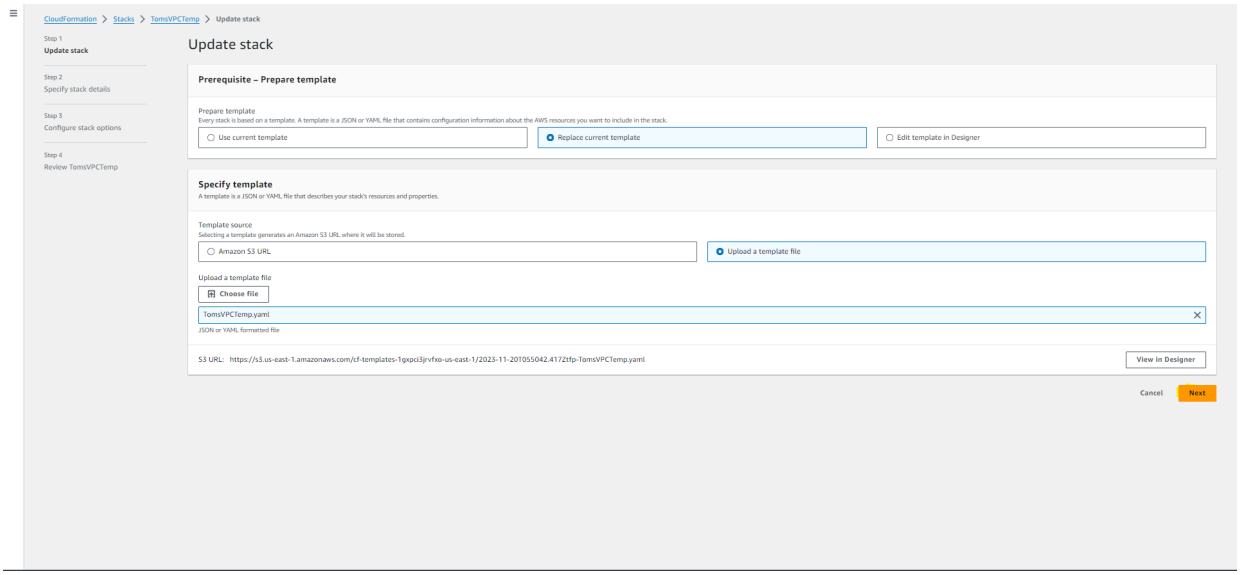
YAML RDS Database
VPC Subnets
all those resources take screenshots and put them there
- there is a large list routing table, “but yeah put 405 few of them”

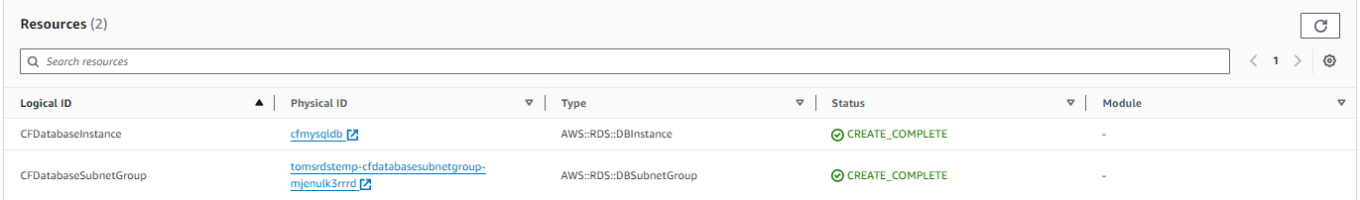
d) Update a resource

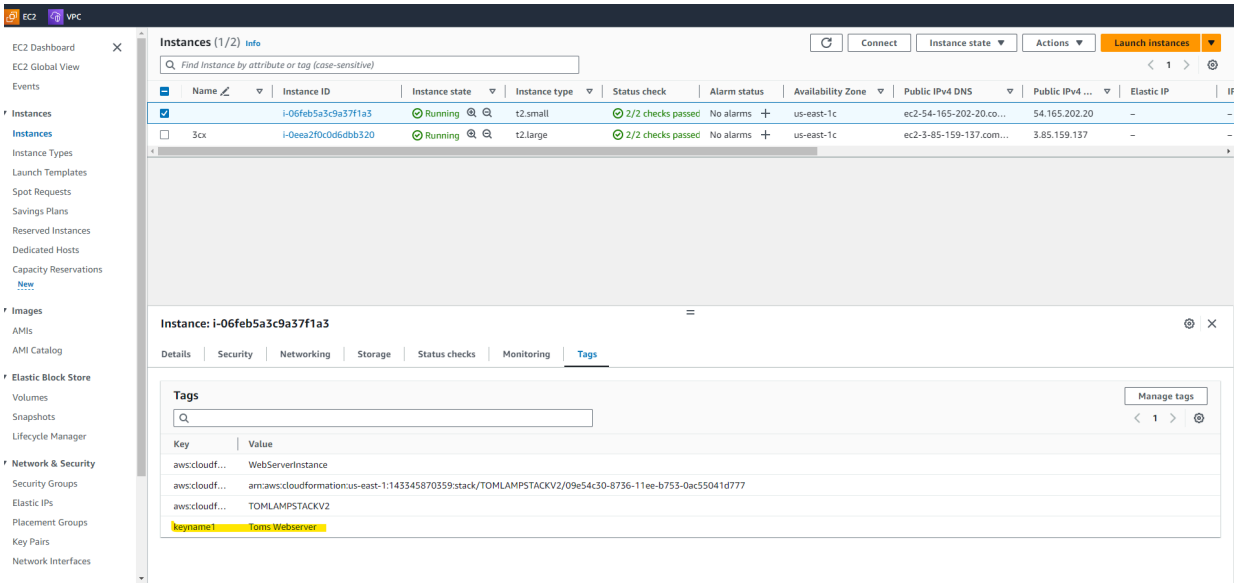
Put additional piece of code inside YAML file to update the resource
i.e tagging instances and resources inside, change resource type and put additional code (like ec1)?



```
            - - '$(mysql '
              - !Ref DBName
              - ' -u root --password='''
              - !Ref DBRootPassword
              - ''' >/dev/null 2>&1 </dev/null); (( $? != 0 ))'
    Properties:
      ImageId: !FindInMap
        - AWSRegionArch2AMI
        - !Ref 'AWS::Region'
        - !FindInMap
          - AWSInstanceType2Arch
          - !Ref InstanceType
          - Arch
      InstanceType: !Ref InstanceType
      Tags:
          - Key: "keyname1"
            Value: "Toms Webserver"
      SecurityGroups:
```







Take screenshot of initial and updated code here so its proof of your update

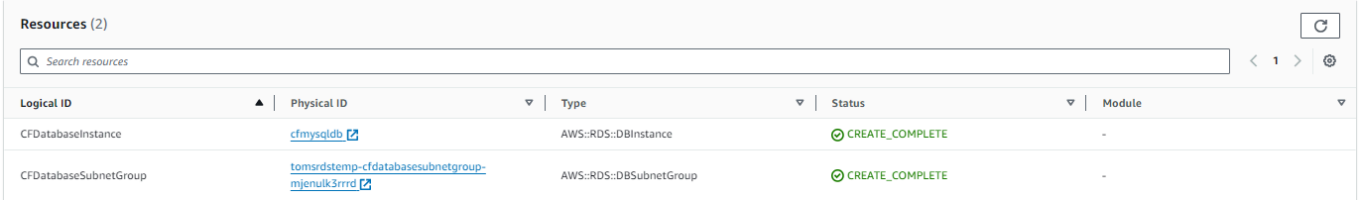
e) Delete a resource

Remove some of the resources, show what piece(s) of code that you removed

Take a screenshot before deleting and after deleting resource

Before

After



*TOMSUPDATEV4.yaml - Notepad

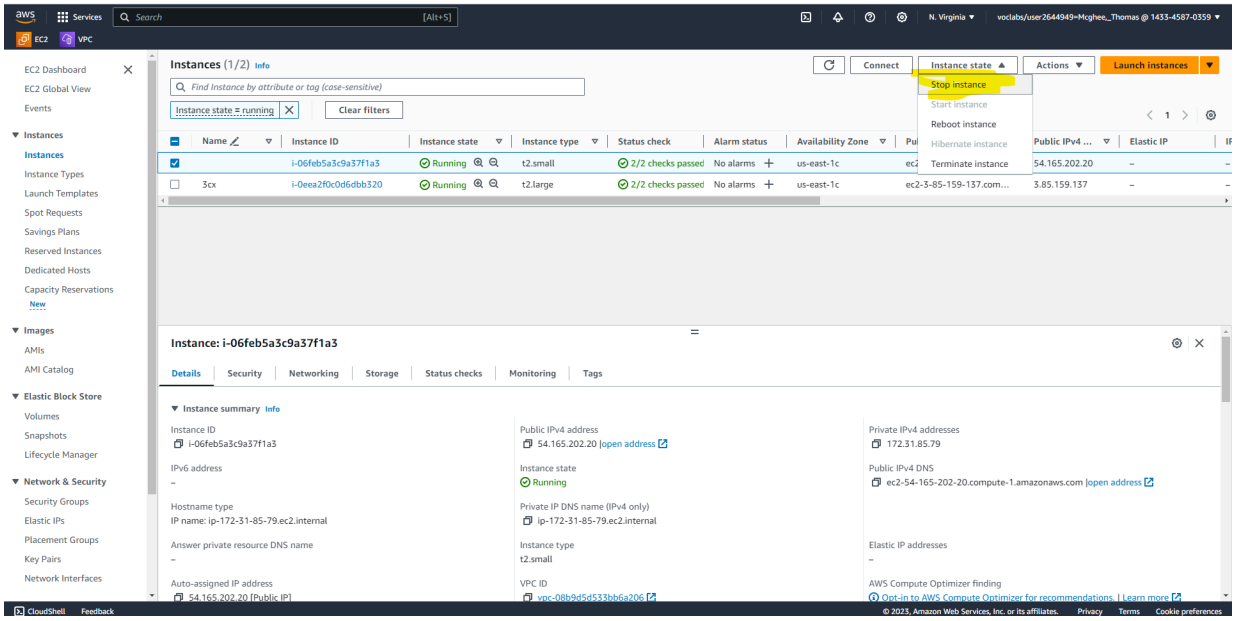
File Edit Format View Help

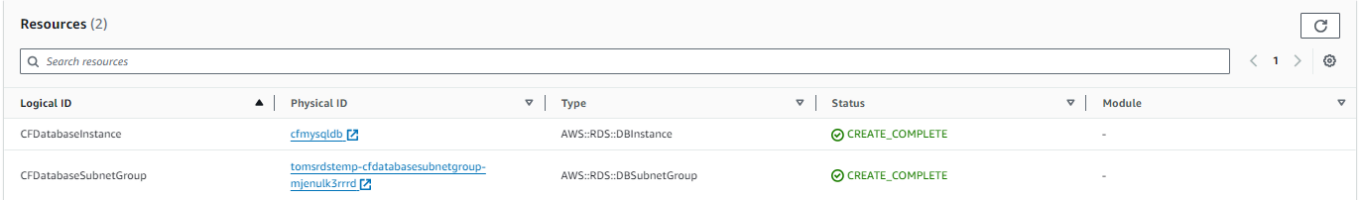
```
Resources:
  WebServerInstance:
    Type: 'AWS::EC2::Instance'
    Metadata:
      'AWS::CloudFormation::Init':
        configSets:
          InstallAndRun:
            - Install
            - Configure
        Install:
          packages:
            yum:
              mysql: []
              mysql-server: []
              mysql-libs: []
              httpd: []
              php: []
              php-mysql: []
          files:
            /var/www/html/index.php:
              content: !Join
                - ''
                - - |
                    <html>
                  - |2
                      <head>
                  - |2
                      <title>AWS CloudFormation PHP Sample</title>
                  - |2
                      <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
                  - |2
                      </head>
                  - |2
                      <body>
                  - |2
                      <h1>Welcome to the AWS CloudFormation PHP Sample</h1>
                  - |2
```

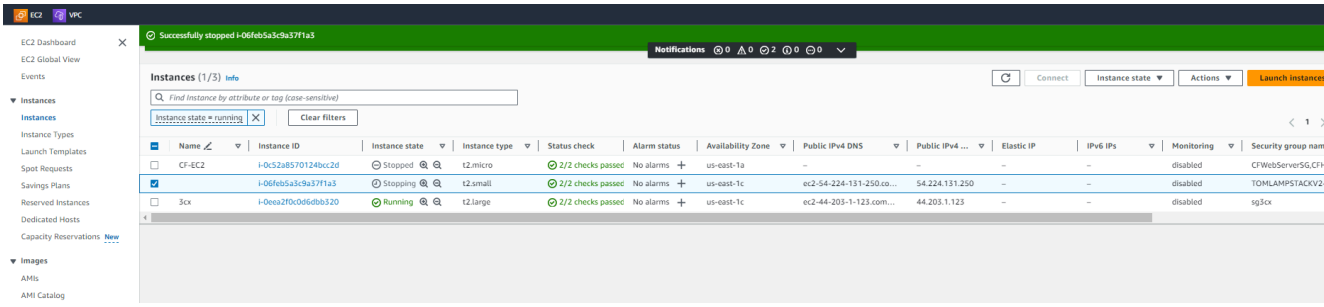
Ln 421, Col 11 100% Windows (CRLF) UTF-8

f) Stop service

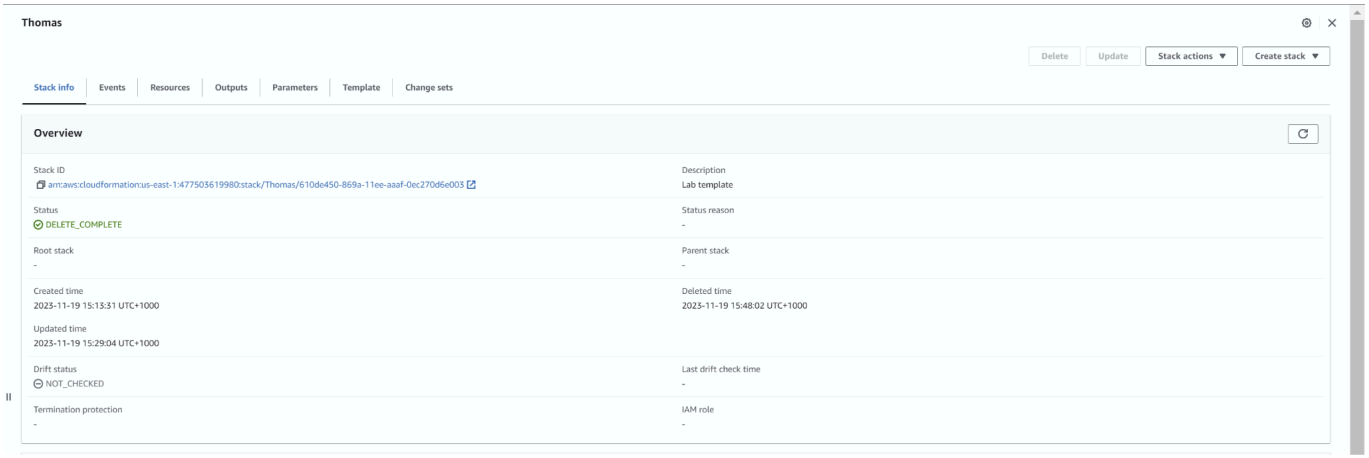
Stop your ec2 instance or whatever instance (like database service)
Take screenshot of instance before and after stopped and put it here
You can use AWS CLI or YAML







Provide screenshots for all the actions performed above as evidence of task completion.



2.3 Test and troubleshot template errors. Document each error and the solution applied. Provide screenshots of the process.

What was the error? FIx line 427: not formatted correctly

Fix: Fix resource by fixing up the spacing and notation behind the code.

How:

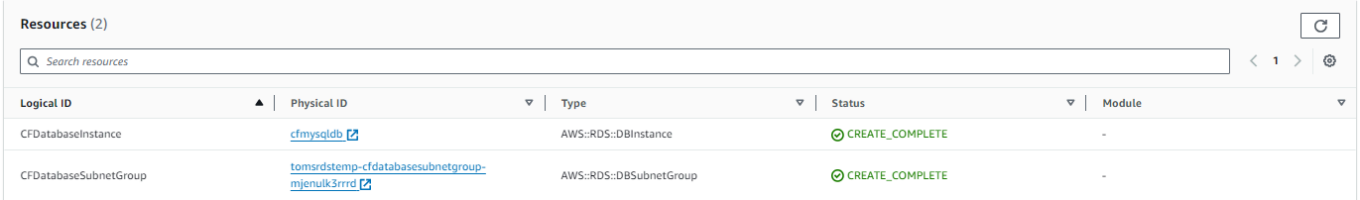
```
Resources:
  S3Bucket:
   Type: 'AWS::S3::Bucket'
    DeletionPolicy: Retain
```

incorrect spacing: here is it fixed:

*TOMSUPDATEV4.yaml - Notepad

File Edit Format View Help

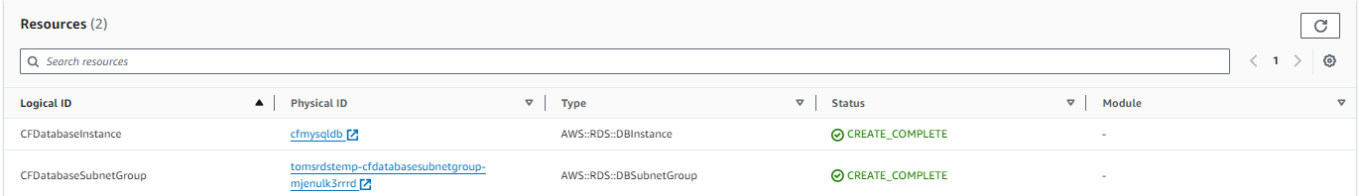
```
Resources:
  S3Bucket:
    Type: 'AWS::S3::Bucket'
    DeletionPolicy: Retain
```



2.4 Create user documentation in the form of a README file that includes what the package is and how to run the containerised application. Summary of those services

Don’t put a step by step guide

md file
Step 1: Review the CloudFormation Template- The Template-lamp-stack.yml CloudFormation template describes the infrastructure setup. It includes definitions for: EC2 instances RDS for MySQL Load balancer Auto-scaling group Security configurations Review the template to understand the resources and their configurations. Step 2: Deploy the CloudFormation Stack Use the AWS Management Console to deploy the stack, click CloudFormation and deploy the stack. Step 3: Monitor Stack Creation Track the progress of stack creation in the AWS CloudFormation Console. Wait for the status to change to CREATE_COMPLETE before proceeding. Step 4: Access and Configure the EC2 instances Once the stack creation is complete, find the public IP addresses of the EC2 instances from the EC2 dashboard. SSH into each instance to configure or inspect the Docker containers as necessary: ssh -i "your-key.pem" ec2-user@instance-public-ip Ensure that the Apache, MySQL, and PHP containers are running correctly. Step 5: Verify the Application Navigate to the DNS name of the load balancer, which you can find in the EC2 console under the Load Balancer section. Verify that the application is serving requests as expected. Step 6: Make Template- Updates (Example) To update an aspect of the deployment, e.g., scaling group policies, modify the Template-lamp-stack.yml file as necessary, and use the AWS GUI to edit the template stacks as necessary. Step 7: Clean Up Resources To avoid incurring unnecessary charges, delete the CloudFormation stack once you're done testing: aws cloudformation delete-stack --stack-name ACME-LAMP-Stack



Ensure that all resources are terminated as expected from the console.

Conclusion
Following this step-by-step guide, you have successfully deployed the ACME Corp LAMP Stack application using a containerized environment on AWS. This README serves as a walkthrough of the deployment process, configuration steps, and best practices for managing and updating cloud resources.

Please adjust the steps and details according to your project's exact specifications, file names, parameters, and procedures. Remember, sensitive information, such as database credentials, should never be hard-coded in your templates or scripts.

2.5 Organise a meeting and present the completed infrastructure as code (IaC) solution to your manager or relevant person in the organisation for approval and signoff.

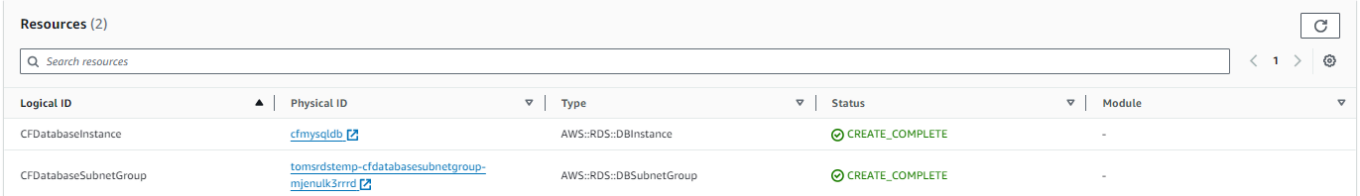
Just fill in the parts described below

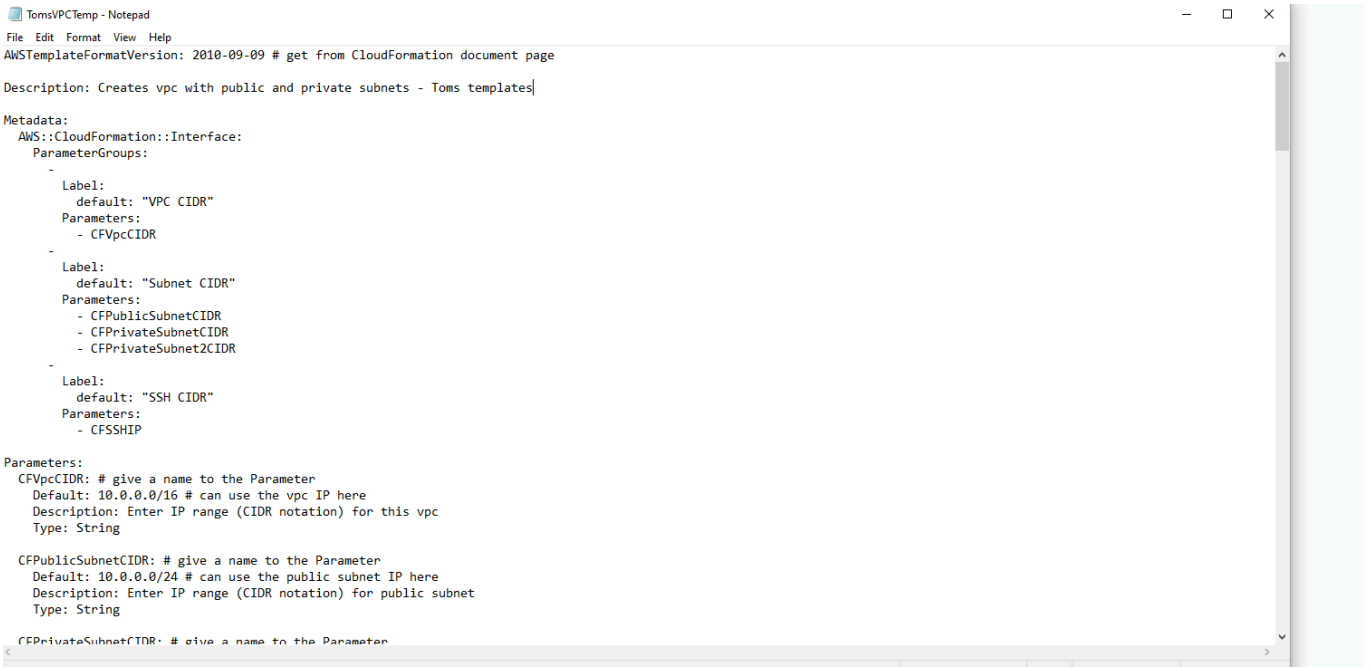
No summary of meeting or any extra information is required

Cloud Infrastructure as code (IaC) – PRE-DEFINED Template SIGNOFF Signing off on this document signifies that the cloud infrastructure as code presented complies with the Client's Business requirements.	
Project Manager or relevant stakeholder Signature: Date:	Web/Cloud Developer Signature: Date:
Documentation NOT APPROVED Please provide feedback on the changes needed.	
APPROVAL ☐ Granted	☐ Not Granted

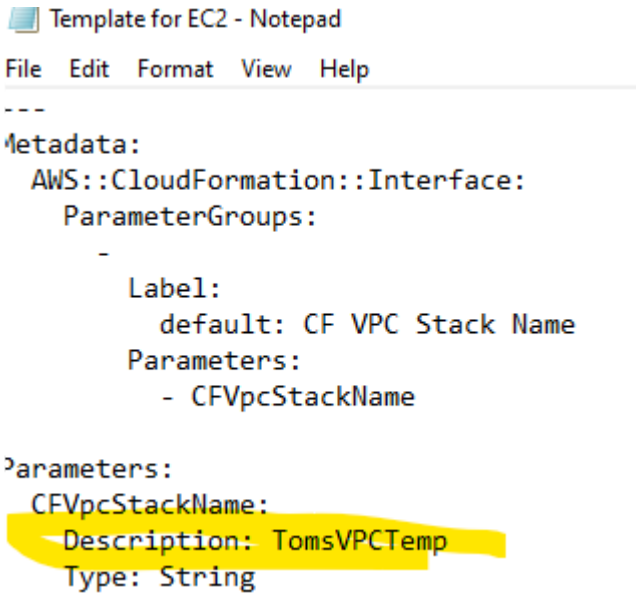
PART 3 - Develop and update infrastructure as code (IaC) templates - Create own template

3.1 Demonstrate knowledge of the selected IaC syntax by creating your own template to deploy a multi-tier application to a cloud orchestration environment for the scenario presented. Ensure that the template pod can manage and run multiple containers and provisions the resources required by the scenario.

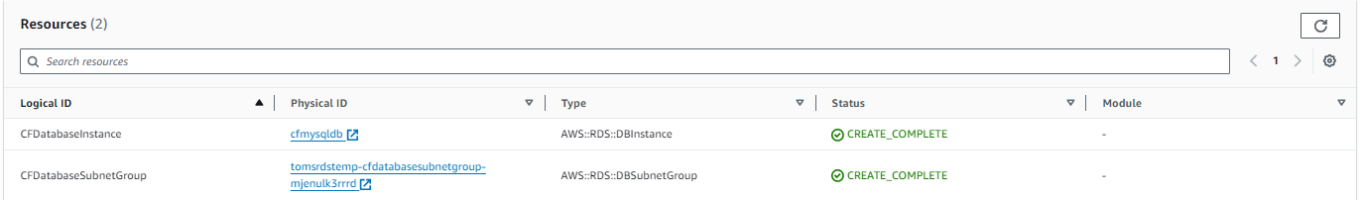


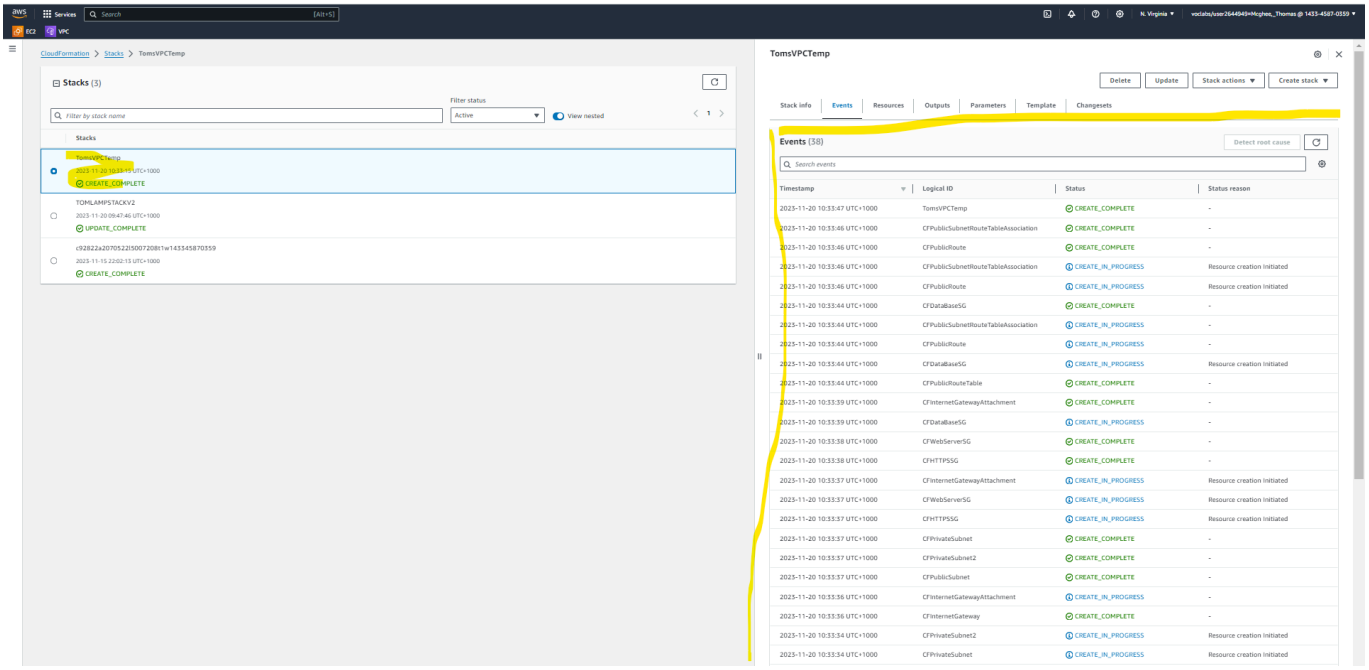


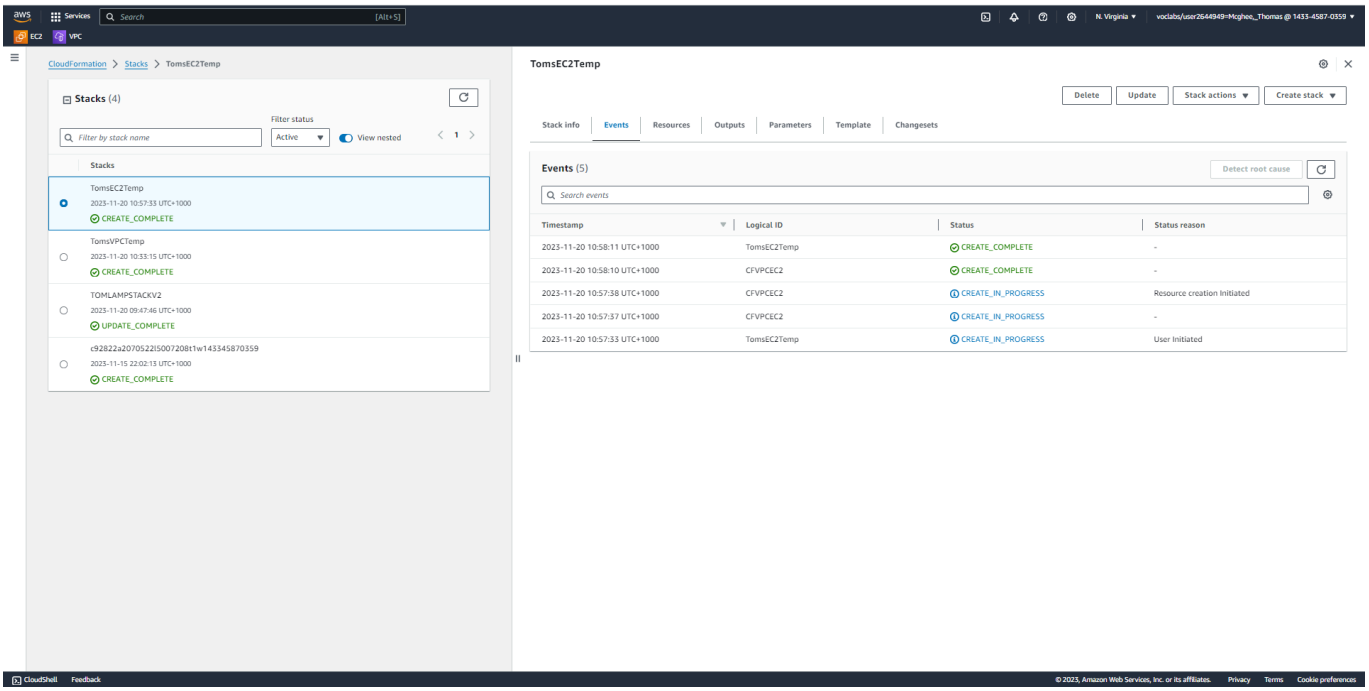
3.2 Configure the template to accept parameters at runtime. Deploy the template. Provide screenshots.

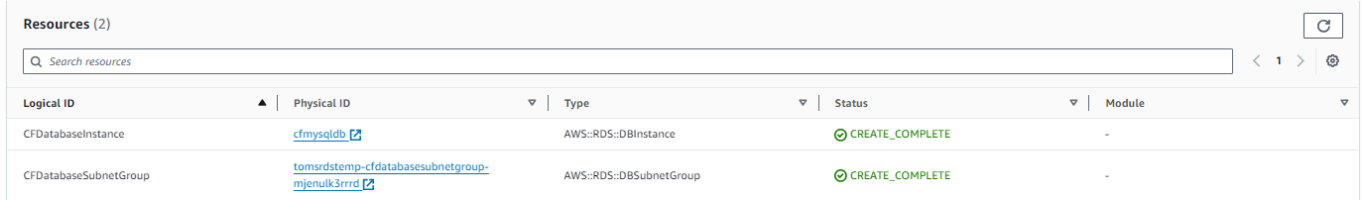


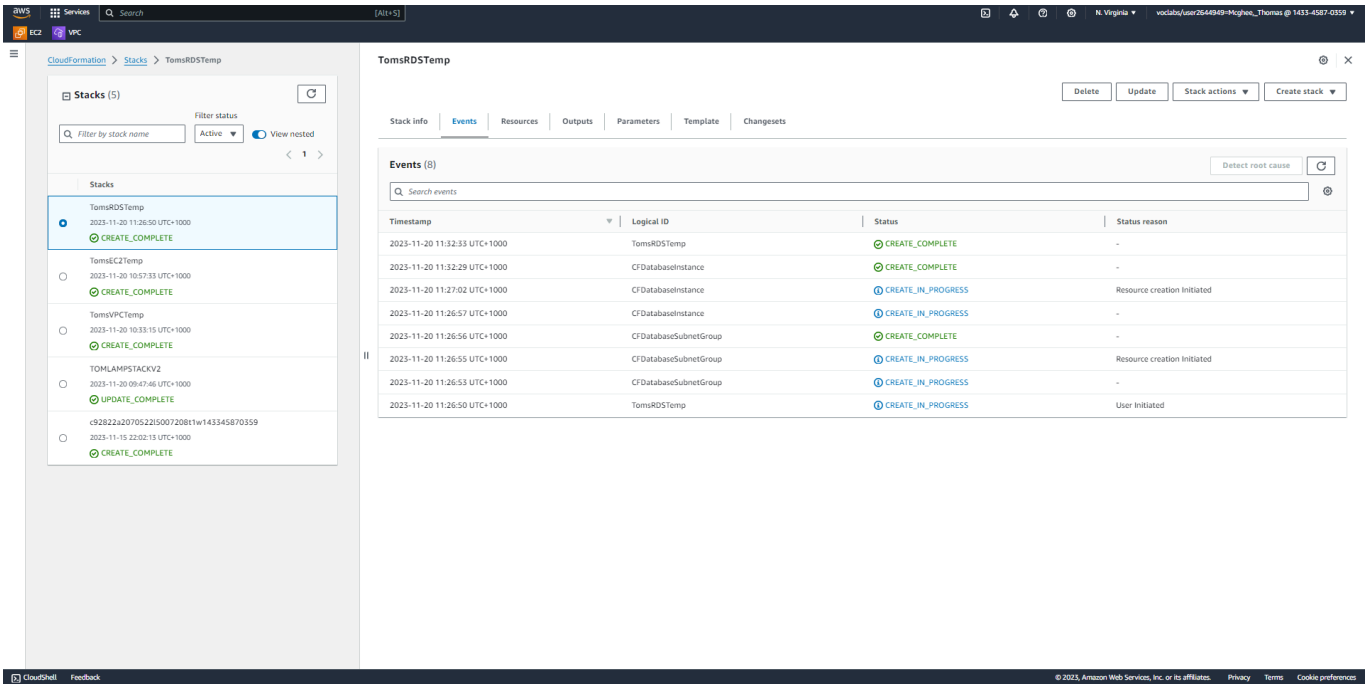
1




CloudFormation > Stacks > TomsVPCTemp
Stacks (3)
TomsVPCTemp
CREATE_COMPLETE
TOMLAMPSTACKV2
UPDATE_COMPLETE
TomsVPCTemp
Delete
Update
Stack actions
Create stack
Stack info
Events
Resources
Outputs
Parameters
Template
Changesets
Events (38)
Timestamp
Logical ID
Status
Status reason


CloudFormation > Stacks > TomsEC2Temp
Stacks (4)
TomsEC2Temp
2023-11-20 10:57:33 UTC+1000
CREATE_COMPLETE
TomsVPCTemp
2023-11-20 10:33:15 UTC+1000
CREATE_COMPLETE
TOMLAMPSTACKV2
2023-11-20 09:47:46 UTC+1000
UPDATE_COMPLETE
TomsEC2Temp
Delete
Update
Stack actions
Create stack
Stack info
Events
Resources
Outputs
Parameters
Template
Changesets
Events (5)
Timestamp
Logical ID
Status
Status reason
2023-11-20 10:58:11 UTC+1000 TomsEC2Temp CREATE_COMPLETE
2023-11-20 10:58:10 UTC+1000 CFVPCEC2 CREATE_COMPLETE
2023-11-20 10:57:38 UTC+1000 CFVPCEC2 CREATE_IN_PROGRESS Resource creation Initiated
2023-11-20 10:57:37 UTC+1000 CFVPCEC2 CREATE_IN_PROGRESS
2023-11-20 10:57:33 UTC+1000 TomsEC2Temp CREATE_IN_PROGRESS User Initiated

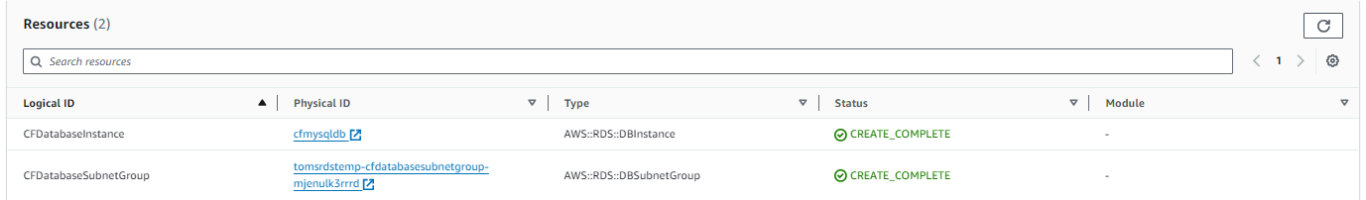

Resources (2)
Search resources
Logical ID
Physical ID
Type
Status
Module
CFDatabaseInstance
cfmysqldb
AWS::RDS::DBInstance
CREATE_COMPLETE
CFDatabaseSubnetGroup
tomsrdstemp-cfdatabasesubnetgroup-mjenulk3rrrd
AWS::RDS::DBSubnetGroup
CREATE_COMPLETE



3.3 Update the template to:

a) Modify a previously deployed resource

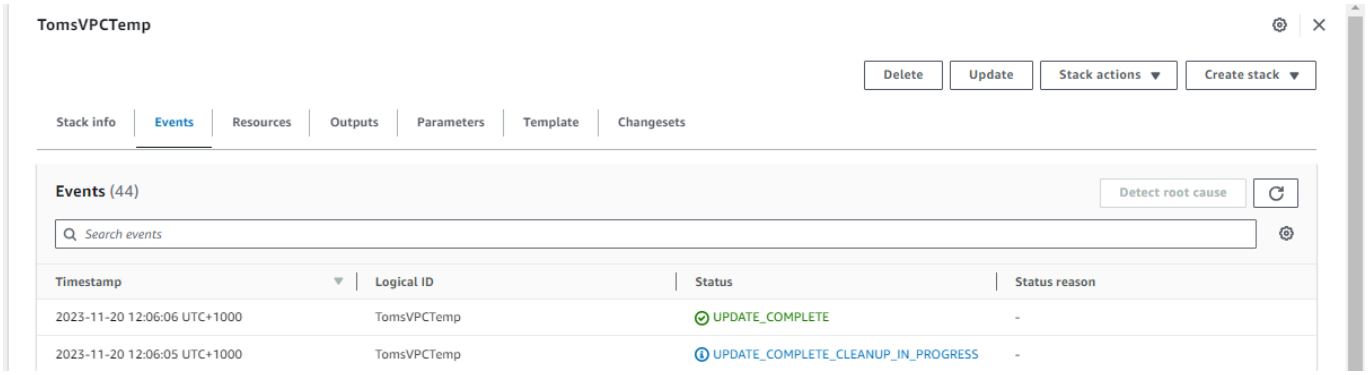
```
---
Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      -
        Label:
          default: TomsVPCTemp
        Parameters:
          - CFVpcStackName

Parameters:
  CFVpcStackName:
    Description: TomsVPCTemp
    Type: String

Resources:
  CFVPCEC2:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: u-24tb1.metal  # 448 vCPUs, 24576GB RAM (~54.86GB RAM per vCPU), uses: Intel Xeon Platinum 8280L (Cascade Lake)
      ImageId: ami-026b57f3c383c2eec  # Default AMI
      KeyName: vockey  # uses lab vockey

      SubnetId:
        Fn::ImportValue: !Sub ${CFVpcStackName}-CFPublicSubnet
      SecurityGroupIds:  # must have a group list, single SG won't work
        - Fn::ImportValue: !Sub ${CFVpcStackName}-CFWebServerSG
        - Fn::ImportValue: !Sub ${CFVpcStackName}-CFHTTPSSG
      Tags:
        - Key: Name
          Value: CF-EC2
      UserData:
        Fn::Base64:
          !Sub |
            #!/bin/bash -xe
            sudo su
            yum update -y
            yum install -y httpd
            yum install -y php
            yum install -y mysql
            cd /var/www/html
            wget https://github.com/jptafe/gorgeous_cupcakes/archive/refs/heads/main.zip
            unzip main.zip
            cp -r jptafe/* /var/www/html/
            rm -rf jptafe main.zip
            systemctl enable httpd
            systemctl start httpd

# may need to restart apache server: 'systemctl stop httpd' and then type 'systemctl start httpd'
# modify database connection: sudo nano /var/www/html/model/data.php
# sudo mysql -h CHANGE-TO-YOUR-DATABASE-URL-HERE -u admin -p
```



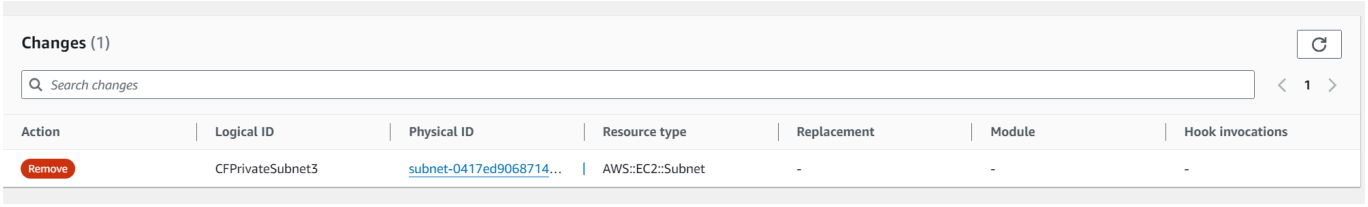
b) Add a new resource

```
CFPrivateSubnet3CIDR:
  Description: '10.0.3.0/24 Private Subnet'
  Type: 'String'
  Default: '10.0.3.0/24'
```

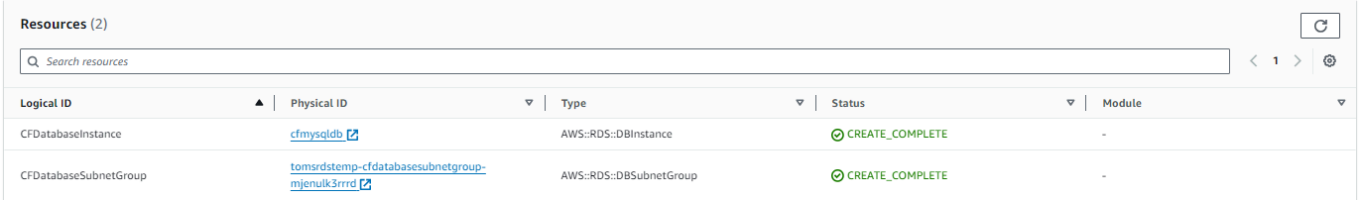
```
CFPrivateSubnet3: # Added updated new private subnet - Useful for extra ec2 instances you want to isolate from eachother, security and seperates attached devices
  Type: 'AWS::EC2::Subnet'
  Properties:
    VpcId: !Ref CFVPC
    CidrBlock: !Ref CFPrivateSubnet3CIDR
    AvailabilityZone: !Select [ 2, !GetAZs '' ]
    Tags:
      - Key: 'Name'
        Value: 'CFPrivateSubnet3'
```



c) Remove a deployed resource



3.4 Use CLI (command line interface) or console to confirm the deployment. Provide screenshots to evidence the actions performed.



Resources (13)

Search resources

Logical ID	Physical ID	Type	Status	Module
CFDataBaseSG	sg-04a6d47fb73dc6d64	AWS::EC2::SecurityGroup	CREATE_COMPLETE	-
CFHTTPSSG	sg-0c6e50e2bb8368bc0	AWS::EC2::SecurityGroup	CREATE_COMPLETE	-
CFInternetGateway	igw-08b454513878cea30	AWS::EC2::InternetGateway	CREATE_COMPLETE	-
CFInternetGatewayAttachment	IGW|vpc-096e7af4262a364fa	AWS::EC2::VPCGatewayAttachment	CREATE_COMPLETE	-
CFPrivateSubnet	subnet-08a8f24cf6a44a8ee	AWS::EC2::Subnet	CREATE_COMPLETE	-
CFPrivateSubnet2	subnet-0991d4a004b060771	AWS::EC2::Subnet	CREATE_COMPLETE	-
CFPrivateSubnet3	subnet-0417ed90687142600	AWS::EC2::Subnet	CREATE_COMPLETE	-
CFPublicRoute	rtb-08f2c9f38f84df36e|0.0.0.0/0	AWS::EC2::Route	CREATE_COMPLETE	-
CFPublicRouteTable	rtb-08f2c9f38f84df36e	AWS::EC2::RouteTable	CREATE_COMPLETE	-
CFPublicSubnet	subnet-0a75af1168a64b9bd	AWS::EC2::Subnet	CREATE_COMPLETE	-
CFPublicSubnetRouteTableAssociation	rtbassoc-06446425abfb0ea03	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE	-
CFVPC	vpc-096e7af4262a364fa	AWS::EC2::VPC	CREATE_COMPLETE	-
CFWebServerSG	sg-01154404c4bf4b6b4	AWS::EC2::SecurityGroup	CREATE_COMPLETE	-

TomsEC2Temp

Delete | Update | Stack actions | Create stack

Stack info | Events | Resources | Outputs | Parameters | Template | Changesets

Resources (1)

Search resources

Logical ID	Physical ID	Type	Status	Module
CFVPCEC2	i-0c52a8570124bcc2d	AWS::EC2::Instance	CREATE_COMPLETE	-

3.5 Using the CLI or console, modify the template to add or change parameters (e.g., API version, spec).

Added S3 Bucket into EC2 template which can be used to store extra pieces of information.

basic script.py | LAMP template.YAML | untitled | task1.yaml | TomsEC2Temp.yaml | task2.yaml

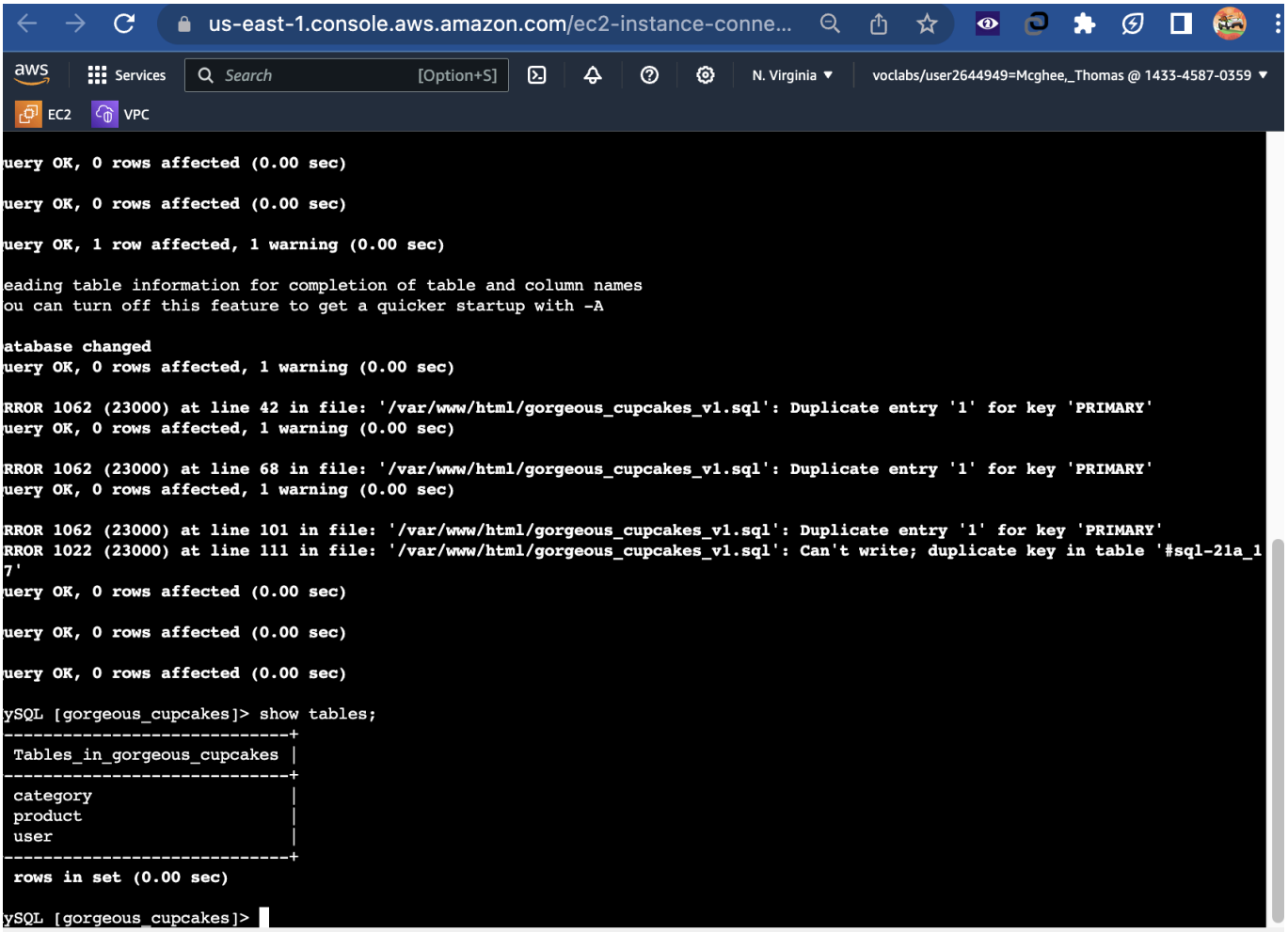
```
---
Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      -
        Label:
          default: TomsVPCTemp
        Parameters:
          - CFVpcStackName

Parameters:
  CFVpcStackName:
    Description: TomsVPCTemp
    Type: String

Resources:

  CFS3BUCKETTOM:
    Type: AWS::S3::Bucket
    Properties:
      BucketName: TomsNewBucket
```

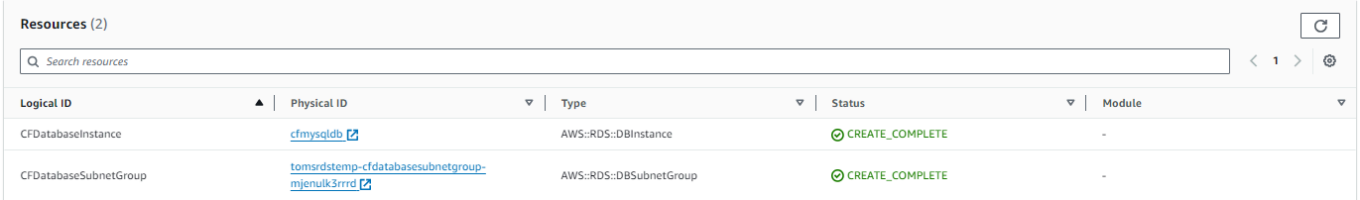
3.6 Create scenario-appropriate secrets using the CLI or configuration file and ensure that it is accessible to the pod created in the template (3.1, 3.2). Provide screenshots of the process.

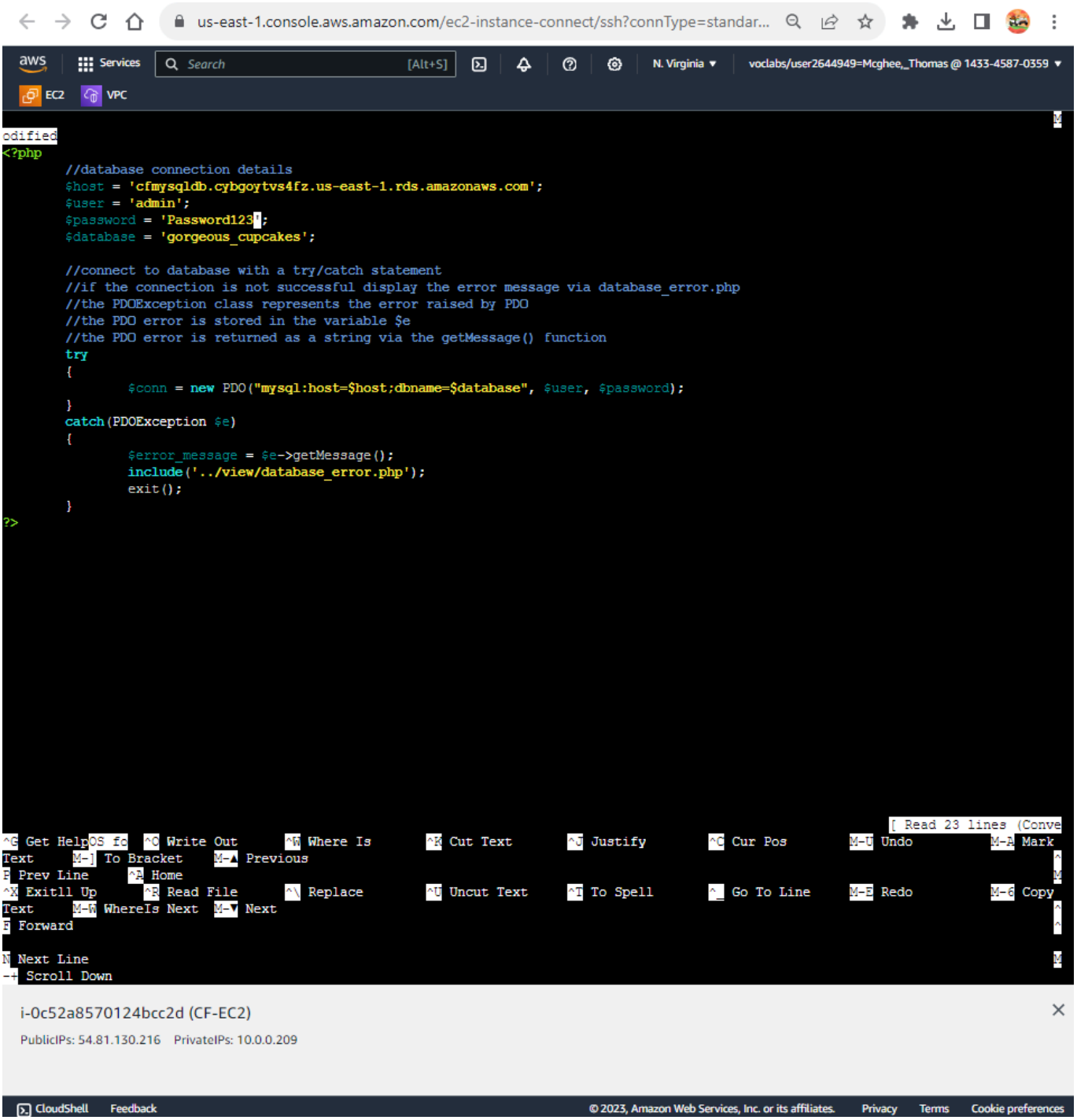
Resources (2)

Search resources

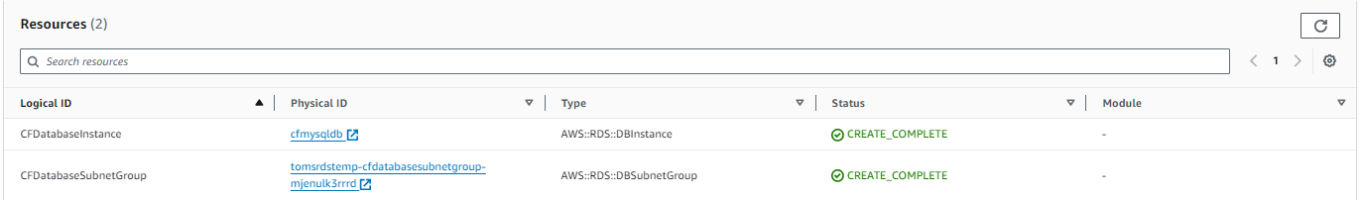
Logical ID	Physical ID	Type	Status	Module
CFDatabaseInstance	cfmysqldb	AWS::RDS::DBInstance	CREATE_COMPLETE	-
CFDatabaseSubnetGroup	tomsrdstemp-cfdatabasesubnetgroup-mjenulk3rrrd	AWS::RDS::DBSubnetGroup	CREATE_COMPLETE	-

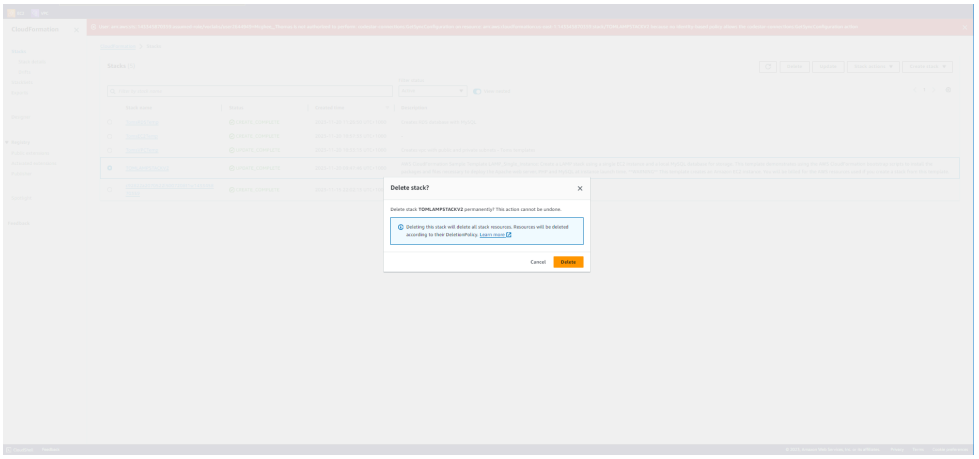

us-east-1.console.aws.amazon.com/ec2-instance-conne...
aws
Services
Search
[Option+S]
N. Virginia
voclabs/user2644949=Mcghee,_Thomas @ 1433-4587-0359
EC2
VPC
uery OK, 0 rows affected (0.00 sec)
uery OK, 0 rows affected (0.00 sec)
uery OK, 1 row affected, 1 warning (0.00 sec)
eading table information for completion of table and column names
ou can turn off this feature to get a quicker startup with -A
atabase changed
uery OK, 0 rows affected, 1 warning (0.00 sec)
RROR 1062 (23000) at line 42 in file: '/var/www/html/gorgeous_cupcakes_v1.sql': Duplicate entry '1' for key 'PRIMARY'
uery OK, 0 rows affected, 1 warning (0.00 sec)
RROR 1062 (23000) at line 68 in file: '/var/www/html/gorgeous_cupcakes_v1.sql': Duplicate entry '1' for key 'PRIMARY'
uery OK, 0 rows affected, 1 warning (0.00 sec)
RROR 1062 (23000) at line 101 in file: '/var/www/html/gorgeous_cupcakes_v1.sql': Duplicate entry '1' for key 'PRIMARY'
RROR 1022 (23000) at line 111 in file: '/var/www/html/gorgeous_cupcakes_v1.sql': Can't write; duplicate key in table '#sql-21a_1
7'
uery OK, 0 rows affected (0.00 sec)
uery OK, 0 rows affected (0.00 sec)
uery OK, 0 rows affected (0.00 sec)
ySQL [gorgeous_cupcakes]> show tables;
Tables_in_gorgeous_cupcakes
category
product
user
rows in set (0.00 sec)
ySQL [gorgeous_cupcakes]>

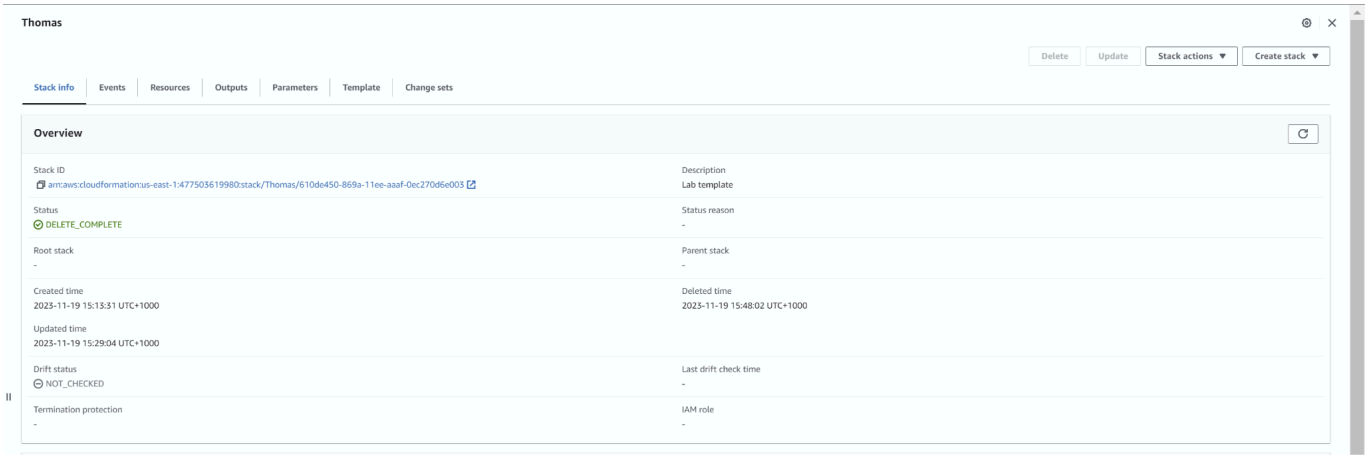

Resources (2)
Search resources
Logical ID
Physical ID
Type
Status
Module
CFDatabaseInstance
cfmysqldb
AWS::RDS::DBInstance
CREATE_COMPLETE
CFDatabaseSubnetGroup
tomsrdstemp-cfdatabasesubnetgroup-mjenulk3rrrd
AWS::RDS::DBSubnetGroup
CREATE_COMPLETE



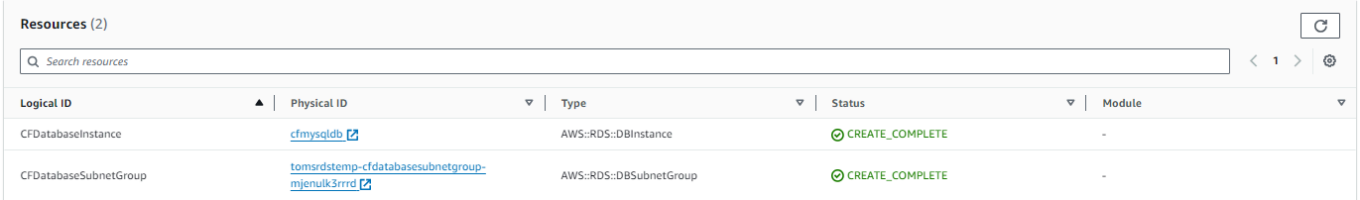
3.7 Delete templates as required. Provide screenshot.

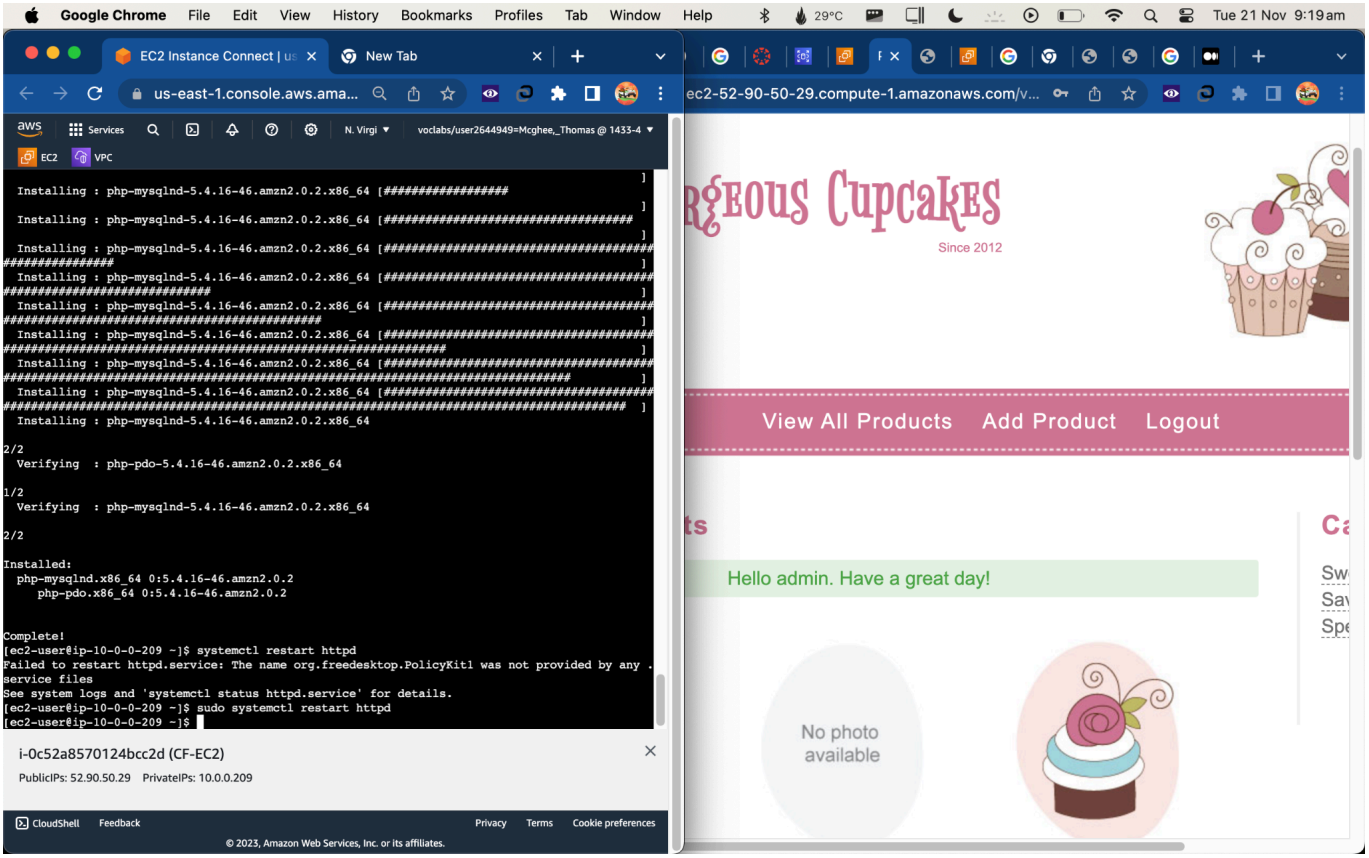




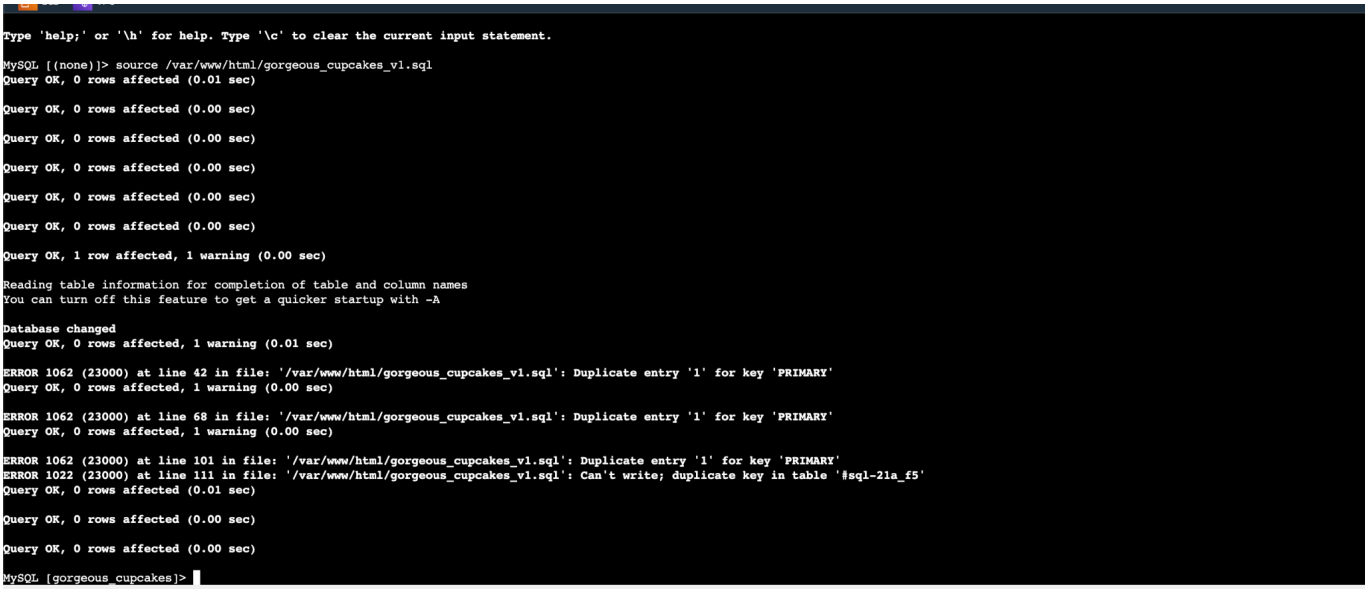


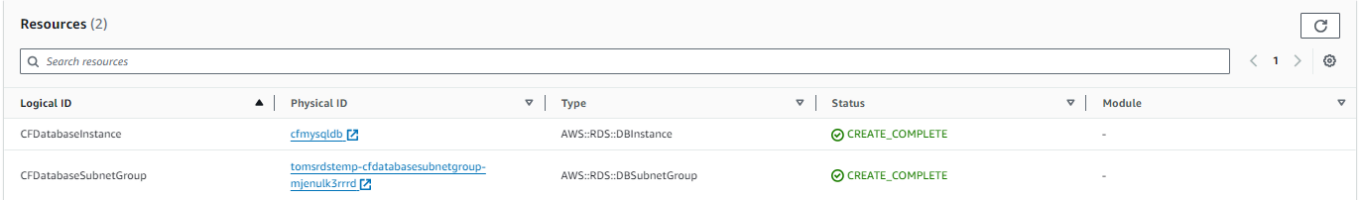
3.8 Test and troubleshot template errors. Document each error and the solution applied. Provide screenshots of the process.





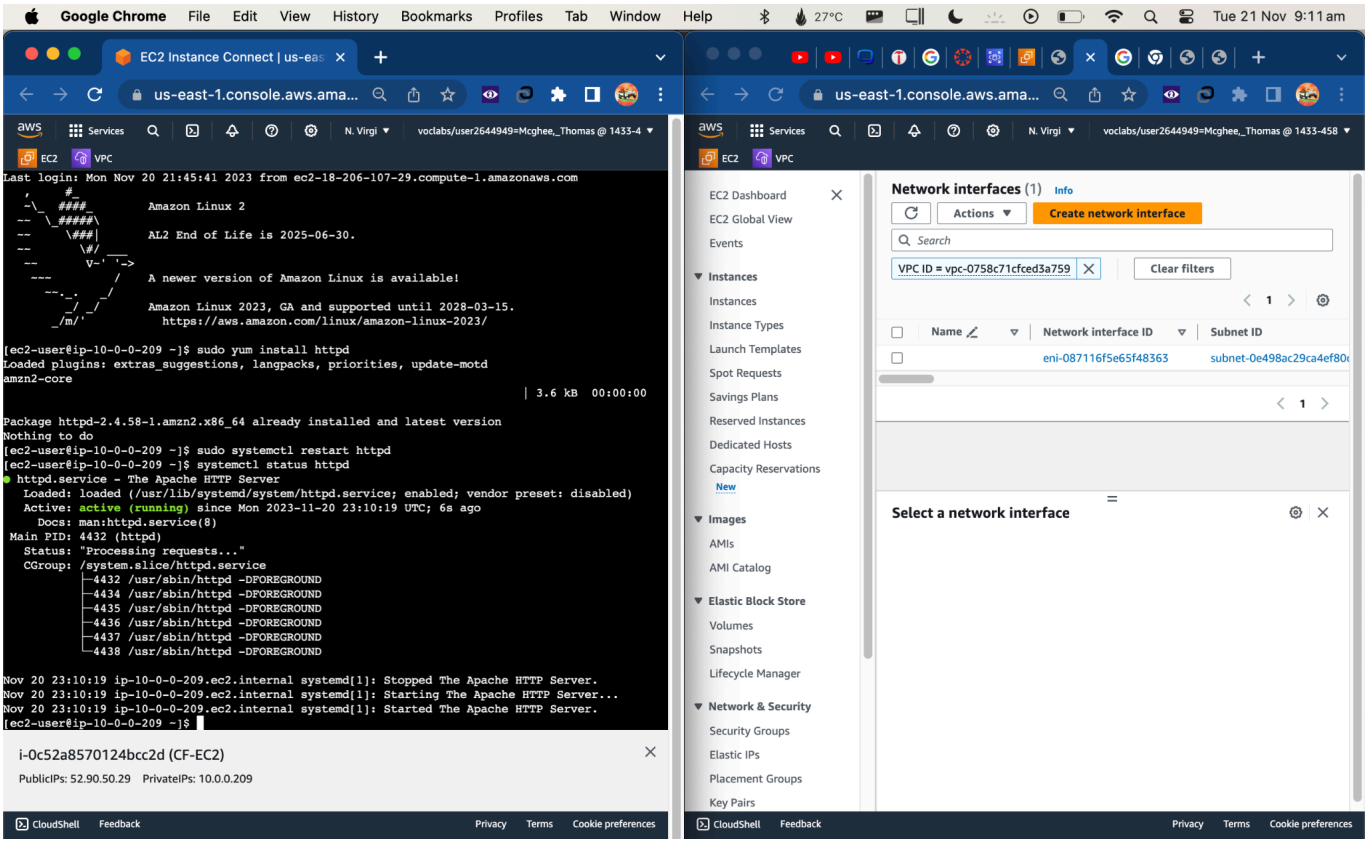
Issues with php and httpd service



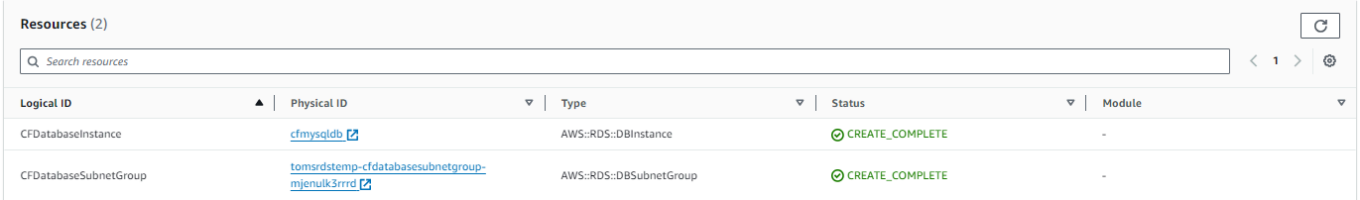


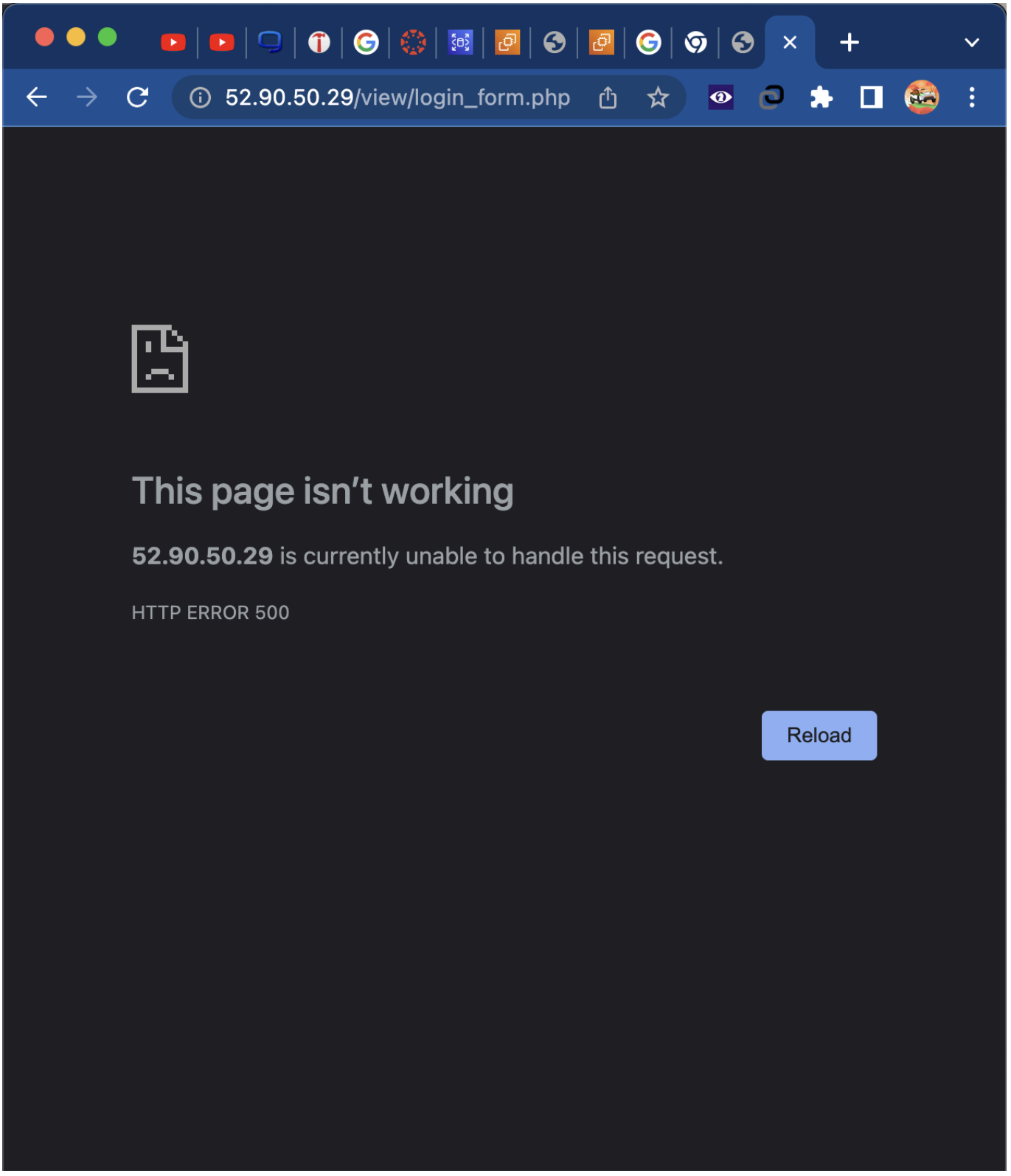
```
MySQL [gorgeous_cupcakes]> exit
Bye
[ec2-user@ip-10-0-0-209 ~]$ systemctl start httpd
Failed to start httpd.service: The name org.freedesktop.PolicyKit1 was not provided by any .service files
See system logs and 'systemctl status httpd.service' for details.
[ec2-user@ip-10-0-0-209 ~]$
```

restarted httpd and checked status using systemcontrol (systemctl)

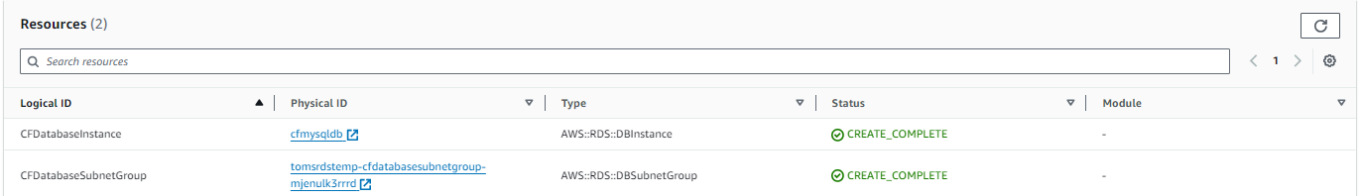


still wasnt working:



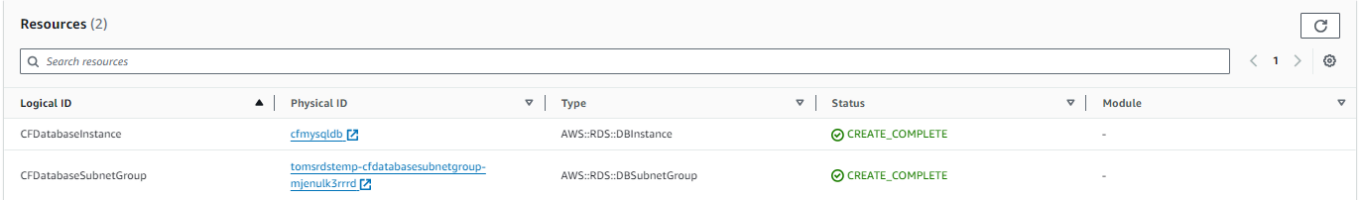


had a look at dependencies used:



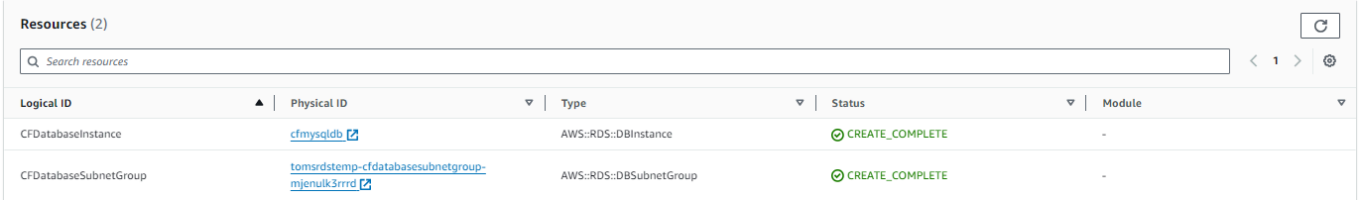
```
Copyright (c) 1997-2013 The PHP Group
Zend Engine v2.4.0, Copyright (c) 1998-2013 Zend Technologies
[ec2-user@ip-10-0-0-209 ~]$ httpd --version
(13)Permission denied: AH00058: Error retrieving pid file /run/httpd/httpd.pid
AH00059: Remove it before continuing if it is corrupted.
[ec2-user@ip-10-0-0-209 ~]$ sudo httpd --version
httpd (pid 4432) already running
[ec2-user@ip-10-0-0-209 ~]$ sudo mysql --version
mysql  Ver 15.1 Distrib 5.5.68-MariaDB, for Linux (x86_64) using readline 5.1
[ec2-user@ip-10-0-0-209 ~]$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                1
On-line CPU(s) list:   0
Thread(s) per core:    1
Core(s) per socket:    1
Socket(s):             1
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 63
Model name:            Intel(R) Xeon(R) CPU E5-2676 v3 @ 2.40GHz
Stepping:              2
CPU MHz:               2400.033
BogoMIPS:              4800.02
Hypervisor vendor:     Xen
Virtualization type:   full
L1d cache:             32K
L1i cache:             32K
L2 cache:              256K
L3 cache:              30720K
NUMA node0 CPU(s):     0
Flags:                 fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 c
lflush mmx fxsr sse sse2 ht syscall nx rdtscp lm constant_tsc rep_good nopl xtopology cpuid ts
c_known_freq pni pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_
timer aes xsave avx f16c rdrand hypervisor lahf_lm abm cpuid_fault invpcid_single pti fsgsbase
 bmi1 avx2 smep bmi2 erms invpcid xsaveopt
[ec2-user@ip-10-0-0-209 ~]$
```

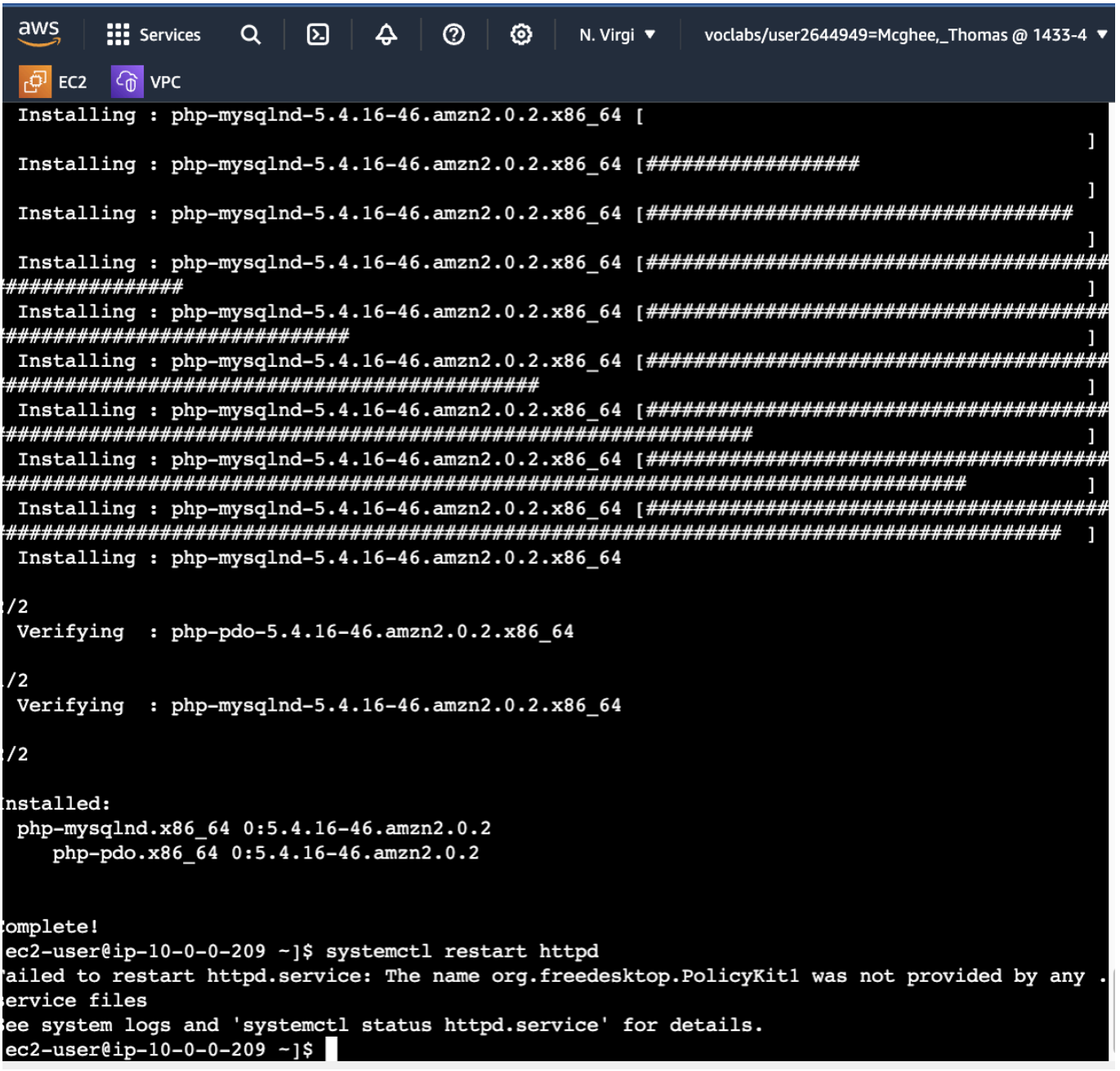
figured out it was a php error looking at httpd error logs:



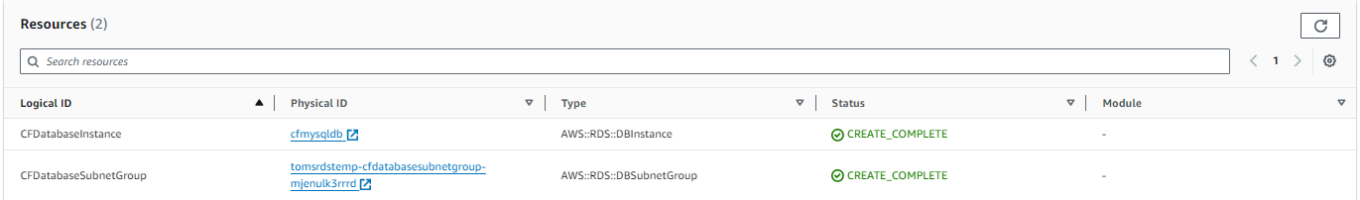
```
 (wrapper: /usr/sbin/suexec)
[Mon Nov 20 21:39:57.223240 2023] [lbmethod_heartbeat:notice] [pid 2958] AH02282: No slotmem f
rom mod_heartmonitor
[Mon Nov 20 21:39:57.223290 2023] [http2:warn] [pid 2958] AH10034: The mpm module (prefork.c)
is not supported by mod_http2. The mpm determines how things are processed in your server. HTT
P/2 has more demands in this regard and the currently selected mpm will just not do. This is a
n advisory warning. Your server will continue to work, but the HTTP/2 protocol will be inactiv
e.
[Mon Nov 20 21:39:57.545488 2023] [mpm_prefork:notice] [pid 2958] AH00163: Apache/2.4.58 () PH
P/5.4.16 configured -- resuming normal operations
[Mon Nov 20 21:39:57.545517 2023] [core:notice] [pid 2958] AH00094: Command line: '/usr/sbin/h
ttpd -D FOREGROUND'
[Mon Nov 20 23:10:18.133662 2023] [mpm_prefork:notice] [pid 2958] AH00170: caught SIGWINCH, sh
utting down gracefully
[Mon Nov 20 23:10:19.212060 2023] [suexec:notice] [pid 4432] AH01232: suEXEC mechanism enabled
 (wrapper: /usr/sbin/suexec)
[Mon Nov 20 23:10:19.239379 2023] [lbmethod_heartbeat:notice] [pid 4432] AH02282: No slotmem f
rom mod_heartmonitor
[Mon Nov 20 23:10:19.239418 2023] [http2:warn] [pid 4432] AH10034: The mpm module (prefork.c)
is not supported by mod_http2. The mpm determines how things are processed in your server. HTT
P/2 has more demands in this regard and the currently selected mpm will just not do. This is a
n advisory warning. Your server will continue to work, but the HTTP/2 protocol will be inactiv
e.
[Mon Nov 20 23:10:19.251827 2023] [mpm_prefork:notice] [pid 4432] AH00163: Apache/2.4.58 () PH
P/5.4.16 configured -- resuming normal operations
[Mon Nov 20 23:10:19.251850 2023] [core:notice] [pid 4432] AH00094: Command line: '/usr/sbin/h
ttpd -D FOREGROUND'
[Mon Nov 20 23:10:43.578433 2023] [:error] [pid 4435] [client 138.44.128.241:45189] PHP Fatal
error:  Class 'PDO' not found in /var/www/html/model/database.php on line 15
[Mon Nov 20 23:10:49.886480 2023] [:error] [pid 4436] [client 131.242.32.4:41640] PHP Fatal er
ror:  Class 'PDO' not found in /var/www/html/model/database.php on line 15, referer: https://w
ww.google.com/
[Mon Nov 20 23:10:52.810957 2023] [:error] [pid 4437] [client 138.44.128.241:42009] PHP Fatal
error:  Class 'PDO' not found in /var/www/html/model/database.php on line 15
[Mon Nov 20 23:10:57.274489 2023] [:error] [pid 4434] [client 138.44.128.241:42165] PHP Fatal
error:  Class 'PDO' not found in /var/www/html/model/database.php on line 15
[Mon Nov 20 23:13:46.297723 2023] [:error] [pid 4483] [client 138.44.128.241:42904] PHP Fatal
error:  Class 'PDO' not found in /var/www/html/model/database.php on line 15
[ec2-user@ip-10-0-0-209 ~]$
```

installed different php dependencies because the current php version wasnt working:



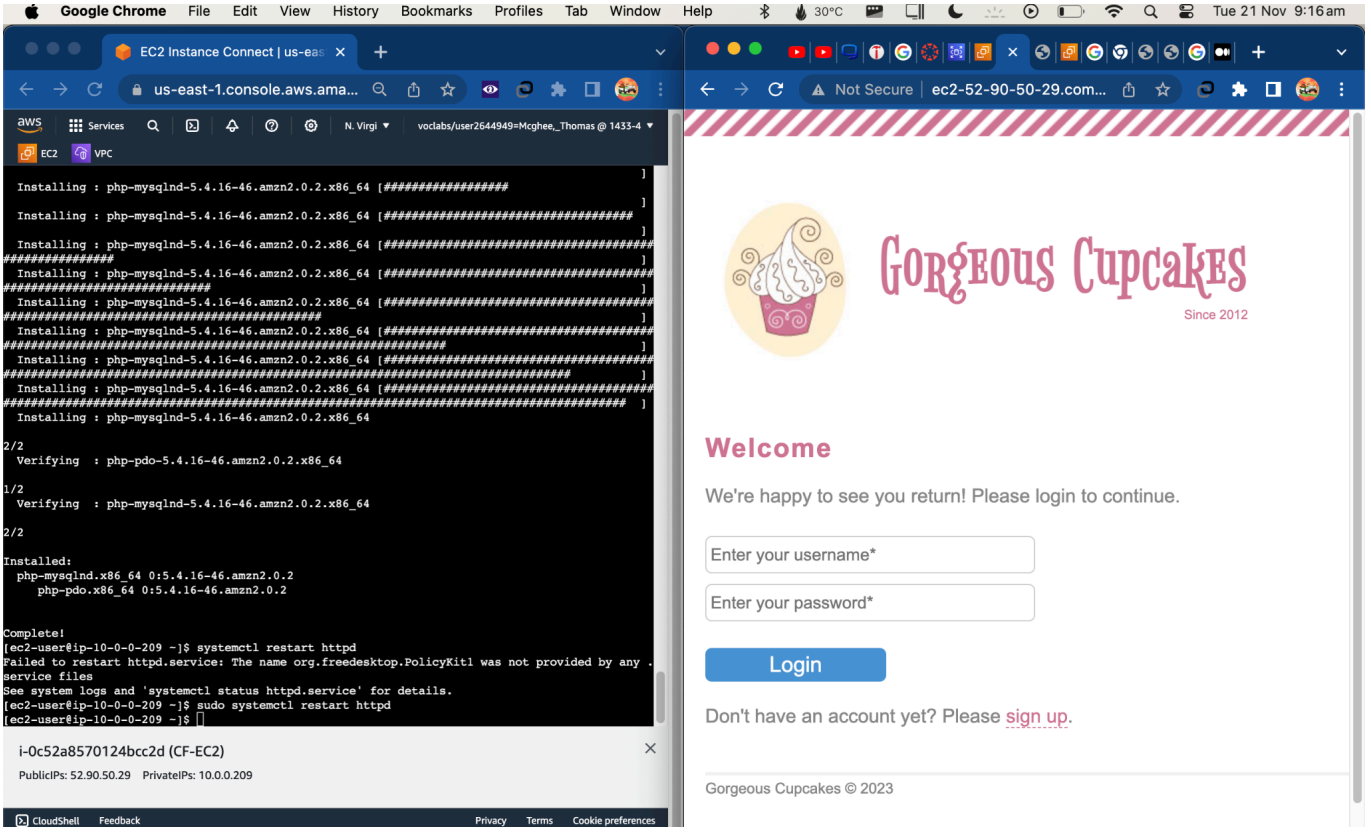


restarted so changes take effect



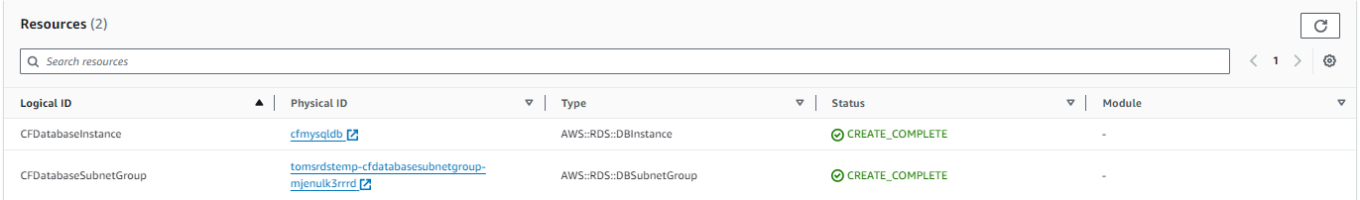
```
  php-mysqlnd.x86_64 0:5.4.16-46.amzn2.0.2
     php-pdo.x86_64 0:5.4.16-46.amzn2.0.2


Complete!
[ec2-user@ip-10-0-0-209 ~]$ systemctl restart httpd
Failed to restart httpd.service: The name org.freedesktop.PolicyKit1 was not provided by any .
service files
See system logs and 'systemctl status httpd.service' for details.
[ec2-user@ip-10-0-0-209 ~]$ sudo systemctl restart httpd
[ec2-user@ip-10-0-0-209 ~]$
```

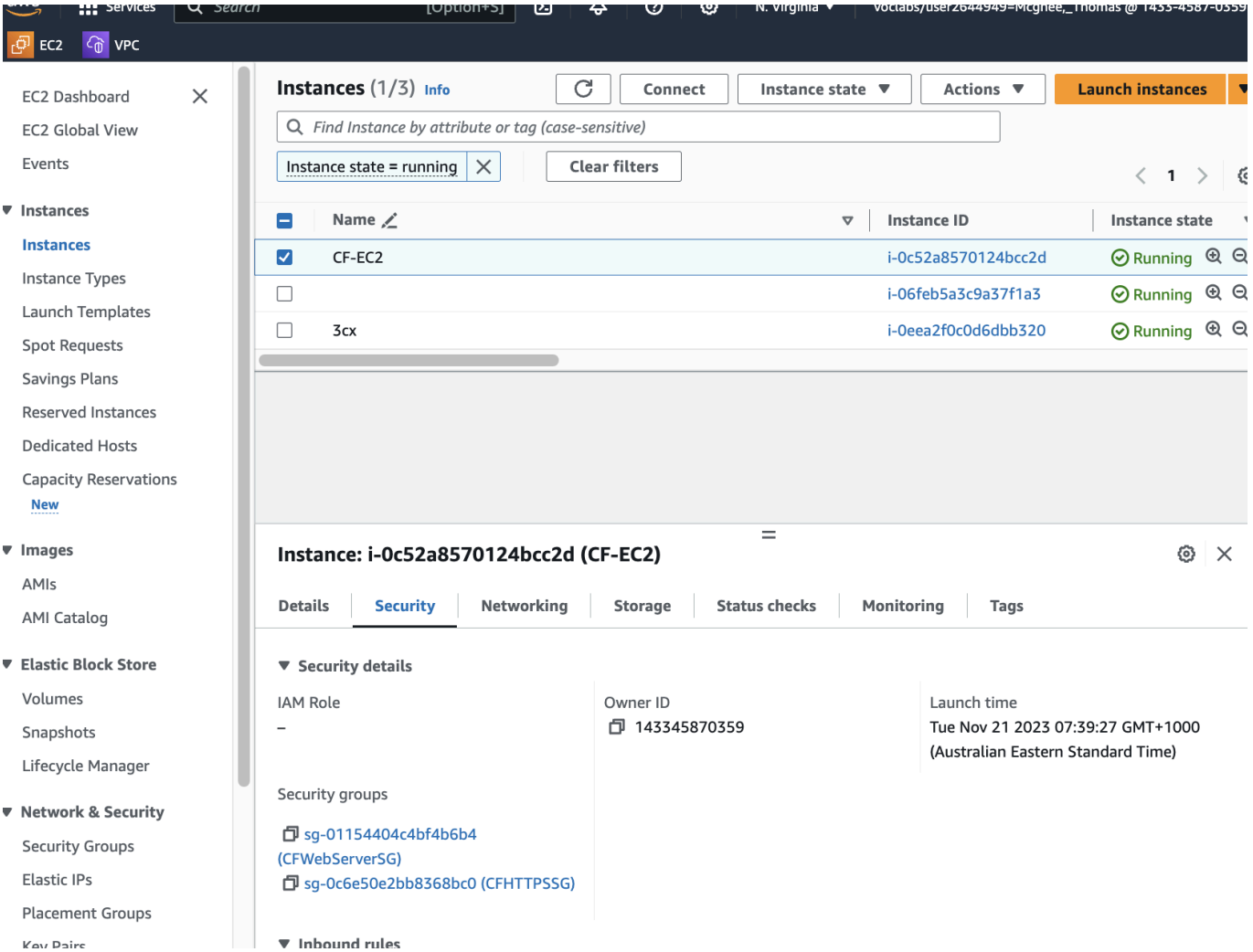
tried connecting again using http

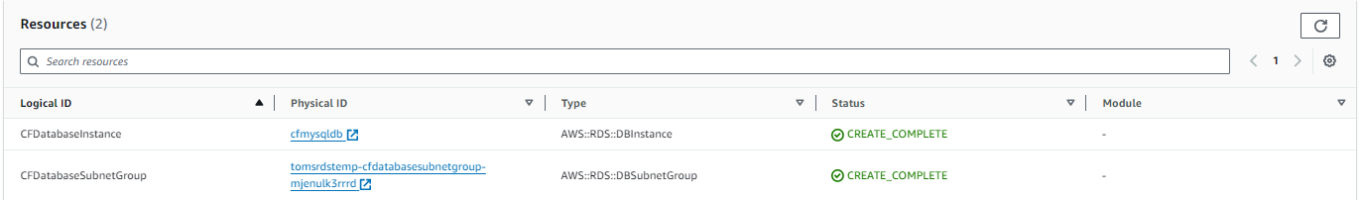


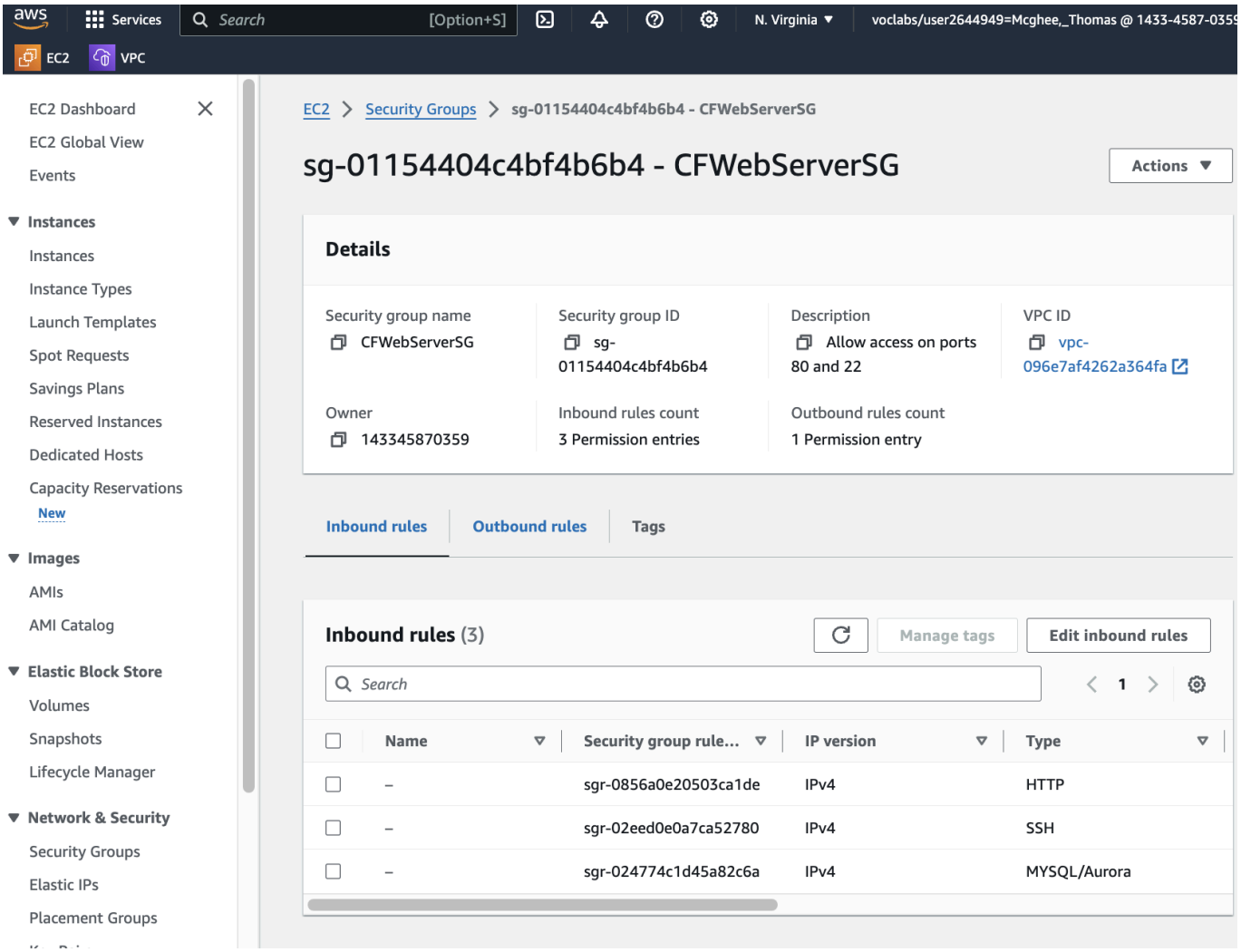
success! :)

Ontop of this, i also made sure that mysql was allowed inside the security group for the ec2 instance, I wasn't sure if this was necessary but i did it anyway just to be safe:




Services
Search
[Option+S]
N. Virginia
EC2
VPC
EC2 Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
New
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Instances (1/3) Info
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
Instance state = running
Clear filters
Name
Instance ID
Instance state
CF-EC2
i-0c52a8570124bcc2d
Running
i-06feb5a3c9a37f1a3
Running
3cx
i-0eea2f0c0d6dbb320
Running
Instance: i-0c52a8570124bcc2d (CF-EC2)
Details
Security
Networking
Storage
Status checks
Monitoring
Tags
Security details
IAM Role
–
Owner ID
143345870359
Launch time
Tue Nov 21 2023 07:39:27 GMT+1000
(Australian Eastern Standard Time)
Security groups
sg-01154404c4bf4b6b4
(CFWebServerSG)
sg-0c6e50e2bb8368bc0 (CFHTTPSSG)
Inbound rules


Resources (2)
Search resources
Logical ID
Physical ID
Type
Status
Module
CFDatabaseInstance
cfmysqldb
AWS::RDS::DBInstance
CREATE_COMPLETE
CFDatabaseSubnetGroup
tomsrdstemp-cfdatabasesubnetgroup-mjenulk3rrrd
AWS::RDS::DBSubnetGroup
CREATE_COMPLETE



3.9 Create user documentation in the form of a README.md file that includes what the package is and how to run the containerised application.

This README outlines the configuration and management of a web application deployed on AWS. The setup includes an Amazon EC2 instance for hosting the web application, an RDS instance for the database, and configured security groups for managing access and security.

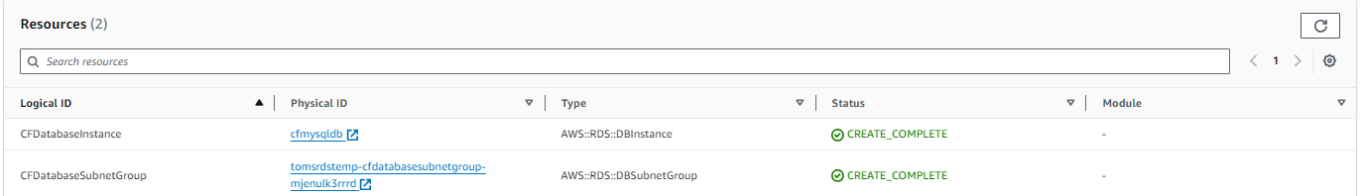
README.md FILE

Components

EC2 Instance: Hosts the web application.

RDS Instance: Manages the application's database.

Security Groups: Control access to both the EC2 and RDS instances.



EC2 Instance Details

Instance ID: i-0c52a8570124bcc2d

Instance Type: t2.micro

Security Groups: CFWebServerSG, CFHTTPSSG

Public DNS/IP: ec2-52-90-50-29.compute-1.amazonaws.com / 52.90.50.29

Key Pair: vockey

Location: us-east-1a

Status: Running, with all health checks passed.

RDS Instance Details

DB Identifier: cfmysqldb

Instance Type: db.t2.micro

Engine: MySQL Community

Endpoint: cfmysqldb.cybgoytvs4fz.us-east-1.rds.amazonaws.com

Port: 3306

VPC: CFvpc (vpc-096e7af4262a364fa)

Security Group: TomsVPCTemp-CFDataBaseSG-12UUSDLSH5LG9 (sg-04a6d47fb73dc6d64)

Publicly Accessible: No

Location: us-east-1a

Security Groups Configuration

CFHTTPSSecurityGroup (sg-0c6e50e2bb8368bc0):

Allows access on port 443 (HTTPS).

Inbound: 2 permission entries.

Outbound: 1 permission entry.

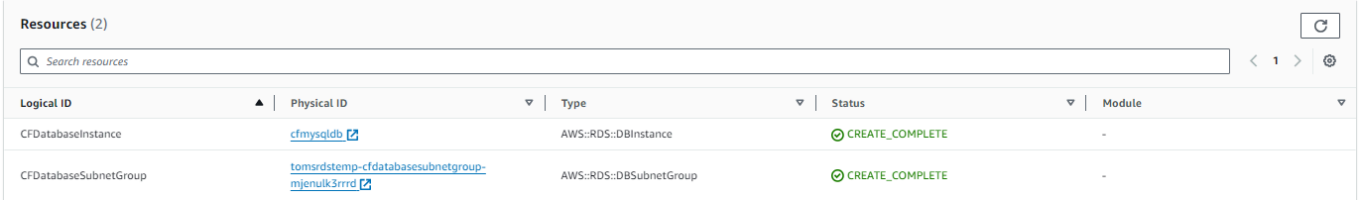
DataBase Security Group (sg-04a6d47fb73dc6d64):

Configured for database access.

Inbound: 1 permission entry.

Outbound: 1 permission entry.

CFSecurityGroup (sg-01154404c4bf4b6b4):



Allows access on ports 80 (HTTP) and 22 (SSH).

Inbound: 3 permission entries.

Outbound: 1 permission entry.

Access and Management

Accessing EC2 Instance

Prerequisites

An AWS EC2 instance with a web server (e.g., Apache) installed.

An AWS RDS instance running MySQL.

Access to the EC2 instance via SSH.

The wget and unzip utilities installed on the EC2 instance.

The MySQL client installed on the EC2 instance for database connection.

Upload and Set Up the Website

1. Copy the Website Files to the Web Server

SSH into your EC2 instance:

bash

Copy code

ssh -i /path/to/your-key.pem ec2-user@your-ec2-public-dns

Navigate to the web root directory:

bash

Copy code

cd /var/www/html

Download the website files:

bash

Copy code

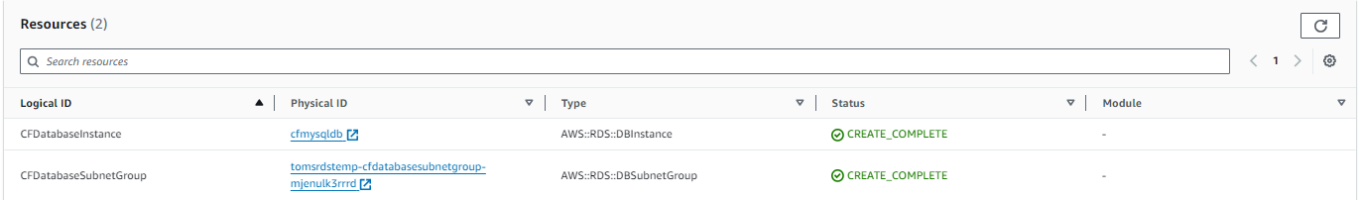
sudo wget https://github.com/jptafe/gorgeous_cupcakes/archive/refs/heads/main.zip

Unzip the downloaded file:

bash

Copy code

sudo unzip main.zip



2. Move the Website Files to the Correct Directory

Change to the directory containing the unzipped files:

bash

Copy code

cd gorgeous_cupcakes-main

Copy all files to the web root:

bash

Copy code

sudo cp -r * /var/www/html/

3. Set Up the Database

Connect to your RDS database:

bash

Copy code

sudo mysql -h [your-rds-endpoint] -P 3306 -u admin -p

Replace [your-rds-endpoint] with your actual RDS endpoint.

When prompted, enter your RDS database password.

Run the SQL script included in the website files:

sql

Copy code

source gorgeous_cupcakes_v1.sql

Check the data tables:

sql

Copy code

select * from product;

select * from user;

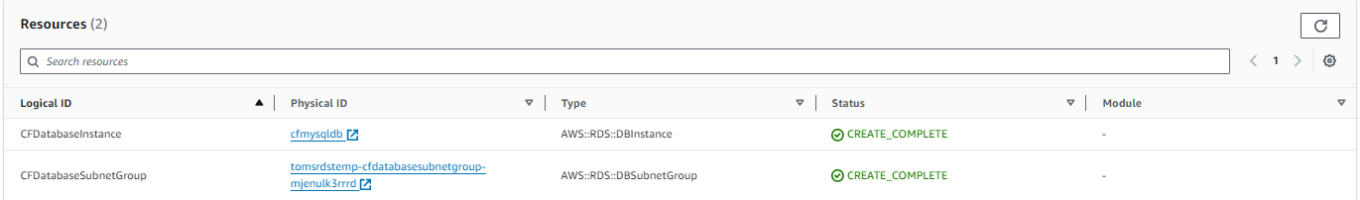
exit

4. Update Database Connection Details

Edit the database.php file in the Model directory:

bash

Copy code



sudo nano ./model/database.php

Update the database connection details:

php

Copy code

$host = '[your-rds-endpoint]';

$user = 'admin';

$password = '[your-rds-password]';

$database = 'gorgeous_cupcakes';

Replace [your-rds-endpoint] and [your-rds-password] with your actual RDS endpoint and password.

Test the Website

Open a Web Browser: Navigate to your EC2 instance's public DNS.

vbnet

Copy code

http://[your-ec2-public-dns]/index.php

Replace [your-ec2-public-dns] with your EC2 instance's public DNS.

Login to the Website: Use the following credentials:

Username: admin

Password: password

Best Practices and Recommendations

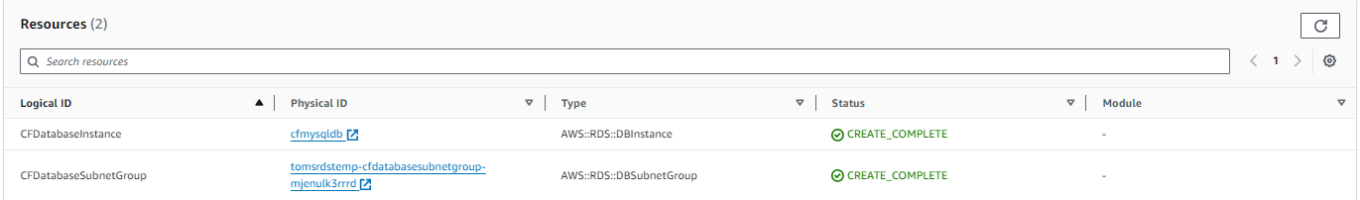
Regularly update and patch your EC2 instance.

Monitor RDS performance and optimize as needed.

Review and update security group rules to ensure minimal access in line with the principle of least privilege.

Troubleshooting

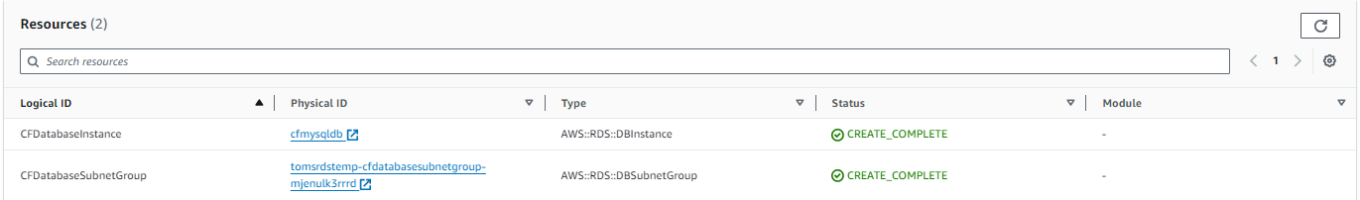
For EC2-related issues, check the instance status checks and system logs.



For RDS issues, review the RDS dashboard and logs for any error messages or alerts.

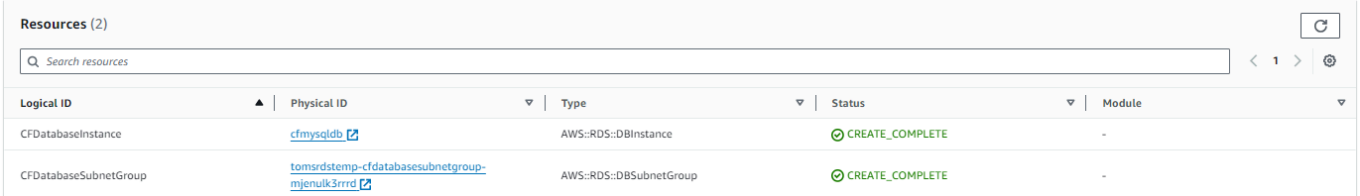
Ensure network accessibility if there are connectivity issues.

End of Readme.md file, if you have any more issues please contact AWS directly here:https://aws.amazon.com/contact-us/



3.10 Organise a meeting and present the completed infrastructure as code (IaC) solution to your manager or relevant person in the organisation for approval and signoff.

Cloud Infrastructure as code (IaC) - Template Development SIGNOFF Signing off on this document signifies that the cloud infrastructure as code presented complies with the Client's Business requirements.	
Project Manager or relevant stakeholder: Elon Musk Signature:  Date: 10/10/2023	Web/Cloud Developer: Tom Signature:  Date: 10/10/2023
Documentation NOT APPROVED Please provide feedback on the changes needed.	
APPROVAL ☐ Granted	☐ Not Granted



PART 4 - Contingency task and cloud-related knowledge concepts

REFERENCING YOUR WORK

Part 4 requires you to carry out research and provide answers to a number of questions. Provide the answers in your own words.

Plagiarism is a form of academic misconduct and will not be tolerated. Include references for all your sources using a formal referencing style such as APA or Harvard.

4.1 **Contingency task:** What happens if the cloud service drops?

Propose three (3) contingency sections that should be included in a Contingency Plan for cloud services. Outline the main steps for each section.

Risk Assessment and Business Impact Analysis (BIA)

Step 1: Identify and classify cloud-based services based on their criticality to the organization's operations.

Step 2: Conduct a thorough risk assessment for each critical service to determine potential vulnerabilities and the probability of service interruption.

Step 3: For each key service, perform BIA to quantify the potential impact of service disruptions on business functions, encompassing financial, reputational, and operational dimensions.

Step 4: Synthesize the risk and impact data to prioritize contingency measures and resource allocation in subsequent sections of the plan.

Service Continuity and Resilience Engineering

Step 1: Design a resilient infrastructure architecture with fault tolerance, redundant systems, and distributed deployment across multiple cloud zones or regions.

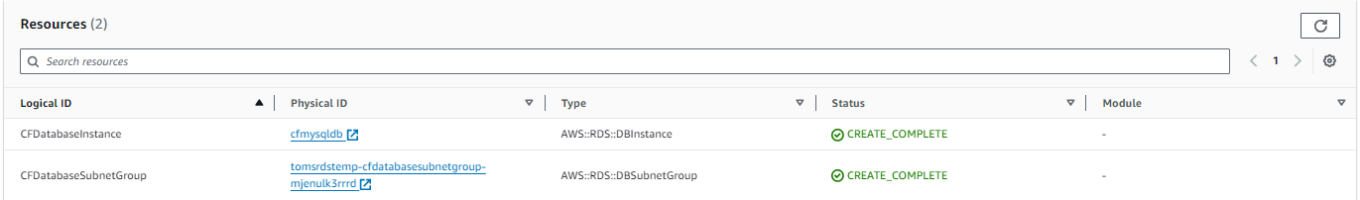
Step 2: Implement rigorous change management and quality assurance processes that include testing for fault tolerance and automatic failover mechanisms.

Step 3: Establish service continuity protocols, including tiered backup strategies and the delineation of both hot and cold recovery paths, ensuring minimal service disruption.

Step 4: Regularly rehearse service restoration procedures through simulated outages to refine the response and reduce recovery time objectives (RTO).

Incident Response and Adaptive Recovery Framework

Step 1: Develop a sophisticated incident response plan with clear escalation paths and



communication protocols that conform to industry best practices such as the NIST Cybersecurity Framework.

Step 2: Incorporate an adaptive recovery framework that employs continuous monitoring and data analytics to enable swift identification of anomalies, allowing for incremental adjustments to the response strategy during an actual incident.

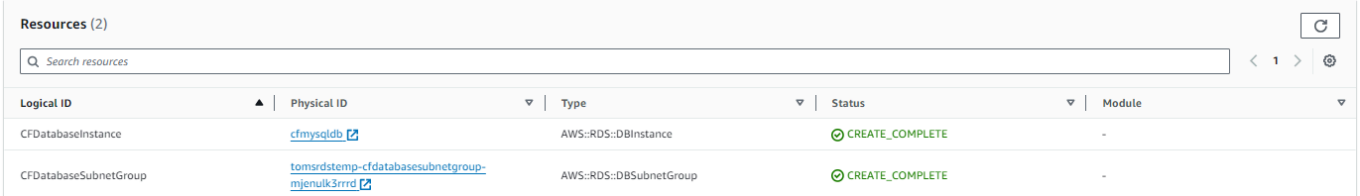
Step 3: Engage in dynamic resource reallocation, leveraging the elasticity of cloud resources to maintain service levels or mitigate service degradation during cloud service failure.

Step 4: Post-incident, conduct a forensic analysis to discern the root cause, draw lessons learned, and iterate the contingency plan accordingly.

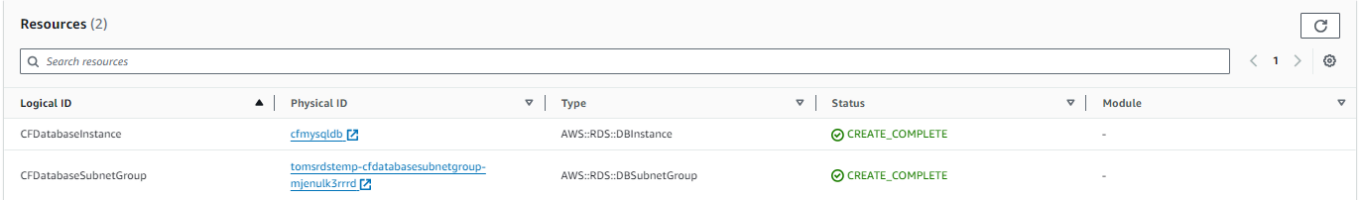
4.2 Select three (3) from the following list of infrastructure as code tools or technologies used in cloud platforms and compare them in terms of features, compatibility with cloud platforms and ease of learning and using the tool.

- AWS CloudFormation
- Azure Resource Manager
- Pulumi
- Terrafom
- Ansible
- Chef
- Puppet
- Docker
- Podman

TECHNOLOGY/TOOL	FEATURES	COMPATIBILITY	EASE of LEARN/USE
AWS Cloudformation	Automates the setup and deployment of AWS resources- Uses templates in JSON or YAML format- Integrates with most AWS	Exclusively compatible with AWS services- Limited support for third-party services or integration	Moderate to learn due to AWS-specific knowledge requirement- Templates can be complex for beginners



	services- Supports rollback and resource tracking- Enables management of infrastructure as code		
Azure Resource Manager	Deploys, manages, and organizes Azure resources- Template-based deployment using Azure Resource Manager templates (ARM templates)- Role-Based Access Control (RBAC) for resource access management- Integration with other Azure services	Designed specifically for Azure cloud platform- Integrates seamlessly with Azure services and features- Limited direct compatibility with other cloud platforms, although hybrid solutions can be engineered	Moderate learning curve, similar to AWS CloudFormation- Familiarity with Azure environment is beneficial
Docker	Containerization platform for developing, shipping, and running applications- Lightweight, standalone, executable package of software- Ensures consistency across multiple development and release cycles- Integrates with CI/CD pipelines	Compatible with most cloud platforms including AWS, Azure, GCP, and others- Extensive third-party integrations and support- Works on various operating systems and environments	Relatively easy to learn for beginners, especially those familiar with command-line interfaces- Strong community and documentation support

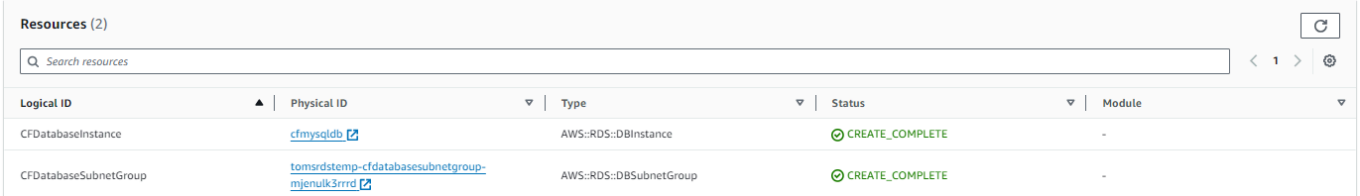


4.3 Research international technology standards applied to cloud computing and select three (3) current ISO standards.

a) For each standard, identify its full name, and publication year and provide a summary of the purpose and scope of the standard.

STANDARD	YEAR	PURPOSE and SCOPE
ISO/IEC 17789: Cloud computing	2014	This standard describes cloud computing systems from four distinct viewpoints: user, functional, implementation, and deployment. It defines four functional layers (user, access, service, and resource) and three roles (cloud service customer, provider, and partner) in the cloud computing ecosystem. The standard aims to provide a comprehensive framework for understanding and discussing cloud computing systems, facilitating a common understanding of its architecture
ISO/IEC 17826: Cloud Data Management Interface (CDMI)	2012	This standard specifies the interface to access cloud storage and manage the data stored therein. It is applicable to developers who are implementing or using cloud storage, providing a standardized interface for cloud data management, which is crucial for interoperability and efficient management of cloud-based data
ISO/IEC 19086-1: Service level agreement (SLA) framework	2016	This standard aims to establish a set of common cloud SLA building blocks, including concepts, terms, definitions, and contexts for creating cloud Service Level Agreements. It provides an overview of cloud SLAs, their relationship with cloud service agreements, and terms commonly used in SLAs. The standard benefits both cloud service providers and customers by facilitating a common understanding and aiding in the comparison of cloud services from different providers. It does not prescribe a standard structure for a cloud SLA or a set of service level objectives, providing flexibility for cloud service providers

b) Would compliance with these standards – or one of the three identified - be applied to the



scenario completed in this portfolio? Justify your answer.

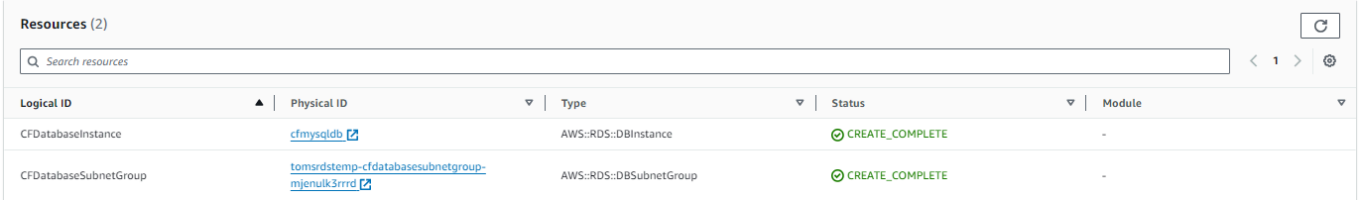
4.4 Outline and explain four (4) industry standards or best practices used for creating, managing, and deploying infrastructure as code (IaC).

Industry standards relating to best practices creating, managing and deploying infrastructure as code span a wide array of protocols, rules and methods, however here are the main four that I've chosen as subjectively the best in this scenario regarding IaC.

1. Version Control: IaC code should always be stored in a version control repository like GitHub as a best practice. Version control grounds specific changes and resources in a catagorized into specific version over time from a main directory, sub directories can be changed and then uploaded to the main branch for release which is essential when working with a large project or multiple groups that are constantly developing on specific IaC templates and infrastructure. A Common type is centralized VCS (which is what i was talking about before).

2. Idempotency: idempotency is the certain property of attain the same set of outcomes provided a certain set of operations is applied without change in continuity. Examples, If you were to run a template of IaC multiple times, it should work the same. In a more abstract sense, assume there is a Monoid, (consists of a set and has binary operation, closed and associative). It has associativity where for example if you were to apply a binary option in this set (say in this terminology applying a cloudformation template). You're applying an operation to the AWS Cloud Infrastructure (the set) and you are manipulating its state.
Regarding associativity, this property is imperative in distributed systems with multiple VPCs, PCs and other properties. For example, if you apply An RDS template first, then a VPC template second, and then EC2 third this should be the same as applying the EC2 template first then applying the RDS instance second and EC2 third. Here is an abstract formula for the set of logic operations i just talked about: $(a + b) + c = a + (b + c)$. The final state of the infrastructure should not change. This practice of IaC is crucial for predictability and reliability of managing infrastructure. Because if it was unpredictable then it wouldn't practically be viable in most scenarios where you need to integrate multiple templates with different components of AWS.
3. Always try to obfuscate hard coded secrets from the direct branch of the IaC template. This practice is to ensure that there is no risk of the organization giving away integral access and authentication to unwanted employees and especially hackers. This can be mitigated by using regex to look for sensitive information inside IaC templates basic example: (\b(password|passwd|pwd)\s*=\s*["'].*?["'])
This practice of finding secrets and password leakage should be used in creating, managing and deploying processes of IaC.
4. Finally and probably the most important: Include proper test environments. (Pre commit hooks)
When you go to deploy an IaC always double check through static code analysis before the code is committed to the repository. Implement test environments and use static code analysis tools as part of your CI/CD pipeline. Pre-commit hooks can ensure that code is checked and tested before it's merged into the main branch, reducing the risk of introducing errors or vulnerabilities into the production environment.

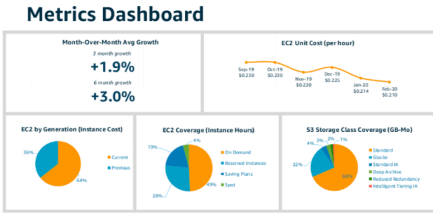
a 4.5 Research and explain at least three (3) metrics currently used in the industry to measure the leverage of using templates to manage and deploy to cloud platforms. The metrics can include



techniques, methods or standards.

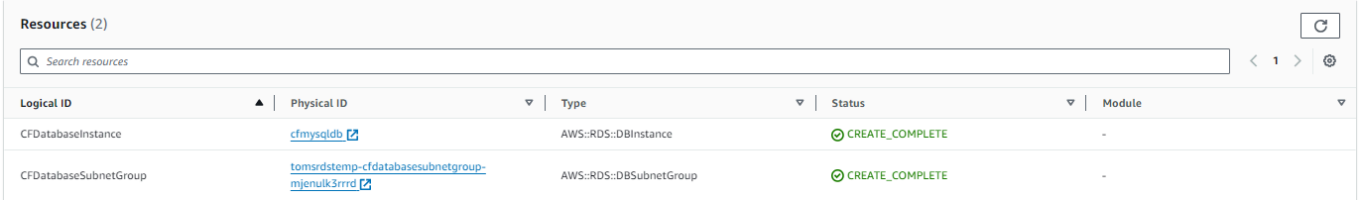
For each metric provide detailed information and examples.

1. Cost optimization metric. Financially viable cloud infrastructure can be hard to find, especially if it's Amazon. If you're not a multi-billion dollar company, then modelling your network around cost optimization metrics can save massive amounts of money that could be better used on other resources. A detailed example of cost optimization using a different component of AWS can be demonstrated with Amazon EC2 instances. Suppose a company is primarily using older EC2 instance generations for its workloads. By analyzing the EC2 Instance Costs metric, they notice that a significant portion of their resources is allocated to these older, less efficient instances. The company then decides to gradually shift to newer instance generations, such as M5 or T3 instances, which offer improved performance at a lower cost. This shift not only enhances computing efficiency but also reduces the overall operational costs, as newer generations are more cost-effective. The company sets a goal to have less than 20% of their EC2 resources on older generations within a year, which is a measurable and actionable objective.



(https://aws.amazon.com/blogs/aws-cloud-financial-management/getting-started-tracking-aws-cost-management-metrics/)

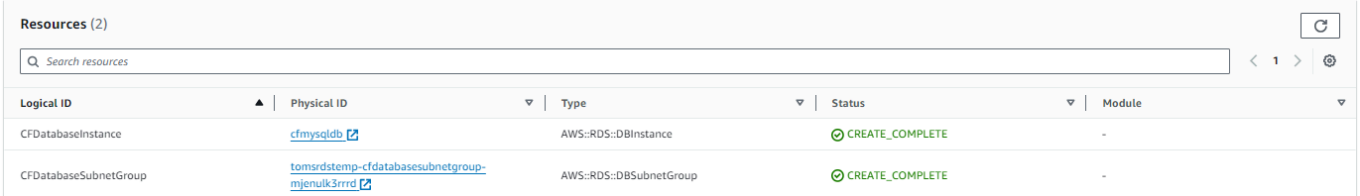
2. Deployment Frequency and Success Rate Metric: The ease and reliability of deployments are important indicators of how well the cloud infrastructure is being managed. Templates can improve these metrics by promoting consistency and repeatability. Example: AWS CloudFormation or Terraform templates help maintain consistent deployment processes across different environments (development, staging, production), which minimizes deployment errors. By using a metric like Deployment Success Rate, the organization can measure the proportion of deployments that succeed without manual intervention or rollback. The metric in action: With the aim to maintain a high Deployment Success Rate, let's say above 99%, the organization can leverage templates that define all required services and resources, ensuring that every deployment has been tested and is known to work. This not only reduces the time developers spend troubleshooting but also leads to more reliable and predictable software releases.
3. Resource Utilization Metric: Efficient use of resources ensures that you are not over-provisioning and hence overpaying for unused capacity. as an example templates enable standardized VM sizes and container configurations, making it easier to match the resource allocation with the actual need. By utilizing a metric like Average CPU Utilization, teams can ensure that their instances are not idling or under heavy stress -- both of which are non-optimal scenarios. Templates with built-in autoscaling policies can dynamically adjust resource allocation in



response to the workload. The metric in action: By using the metric of Resource Utilization Efficiency, the goal could be to maintain the average CPU utilization within an optimal range (e.g., between 50-70%). With templates, when the utilization goes out of this range, scaling actions can be automatically triggered to either scale up or down, optimizing costs.

4.6 Research industry-standard hardware and software products utilised in cloud development and deployment. Select two (2) hardware and two (2) software products and summarise your research in the table below.

PRODUCT	FEATURES/CAPABILITIES	APPLICATION Suitable for	DATA STORAGE
HPE ProLiant Servers	- High processing power, memory, and storage- Advanced security features- Intelligent system tuning- Workload performance optimization	Used in data centers for private and hybrid cloud strategies- Running virtual environments- Supporting various cloud services	Enterprises needing reliable, scalable server hardware for cloud infrastructure
Dell EMC PowerEdge Servers	- Automation of essential server lifecycle management tasks- Robust built-in security- Scalable storage and compute functions- Flexible configurations- Multi-vector cooling	- Supports a range of applications from web hosting to data-intensive tasks like analytics and AI in cloud environments	- Cloud service providers and businesses of all sizes seeking customizable hardware solutions for cloud deployments
VMware vSphere	Virtualization platform- Server consolidation- Business continuity- Simplified management- Application performance enhancement- Robust and resilient for virtualized cloud environments	Creation and management of virtual machines- Essential for cloud computing initiatives- Running multiple virtual servers on a single physical server	Businesses seeking a virtualized environment for data centers- Enables cloud computing and improves resource utilization



Ansible by Red Hat	Open-source automation platform- Software provisioning, configuration management, application deployment- Minimal, consistent, secure, and reliable- Uses YAML (easy-to-understand language)	Automating cloud provisioning, configuration management, application deployment, and intra-service orchestration	Organizations automating cloud development, deployment, and operations- Suitable for those preferring a platform without a steep learning curve

End of Assessment