**Optimizing Web Application Deployment on AWS**

Brandon Luff and Benjamin Gaudlip

Robert Morris University

INFS 4460: Cloud Computing and AWS

Professor Al-Jaroodi

December 12, 2023

Table of Contents

**Lab Account Used: Brandon Luff**

Introduction ……………................................................................................................. 3

1. Client Needs Analysis .................................................................................................. 4

1.1 Background and Current Challenges …................................................................. 4

1.2 Detailed Analysis of Current Website and Database Implementation ................... 4

1.3 Initial Conclusions .................................................................................................. 4

2. Solution Design …………............................................................................................. 5

2.1 AWS Architecture Overview ………...................................................................... 5

2.2 Deployment Strategy ……………………............................................................... 5

2.3 Security Measures ………………........................................................................... 5

2.4 Scalability and High Availability ........................................................................... 6

2.5 Systems Manager Parameter Store ………………………………………………. 6

3. Implementation and Testing ......................................................................................... 6

3.1 Deployment Steps …............................................................................................... 6

3.2 Testing and Validation ........................................................................................... 7

3.3 Screenshots and Descriptions of Key Implementation Steps ................................. 7

4. Cost Analysis ……......................................................................................................... 7

4.1 Breakdown of AWS Service Costs .......................................................................... 7

4.2 Cost Optimization Strategies ................................................................................... 8

4.3 Comparison with Previous Hosting Solution .......................................................... 8

5. Challenges and Solutions ………................................................................................... 8

5.1 Overview of Issues Encountered ………….............................................................. 8

5.2 Problem-Solving Approaches …………….............................................................. 9

5.3 Lessons Learned ……………………....................................................................... 9

6. Conclusions and Future Considerations ……………………………………………… 9

Appendix A ....................................................................................................................... 11

Figures .............................................................................................................................. 12

References ........................................................................................................................ 16

**Introduction**

In an era dominated by cloud computing and online services, the deployment and management of web applications have become pivotal for organizations across various sectors. This report delves into the intricacies of deploying a PHP application on the Amazon Web Services (AWS) platform, tailored to meet the specific needs of the Example Social Research Organization, a fictitious but representative non-profit entity in social science research.

The Example Social Research Organization has reached a critical juncture in its growth trajectory. Initially a brainchild of researcher Shirley Rodriguez, the organization's website has evolved from a modest data-sharing platform to a widely accessed repository of global development statistics. However, this rapid expansion has brought significant challenges, particularly regarding website performance and security. These issues were further compounded by an attempted ransomware attack, underscoring the need for a robust and secure hosting environment.

Therefore, this report presents a comprehensive solution that addresses the current technological bottlenecks and lays a foundation for scalable, secure, and efficient future operations. The proposed solution encompasses deploying the organization's PHP application on an Amazon Elastic Compute Cloud (EC2) instance, creating, and managing a MySQL database, and implementing security measures to protect sensitive data and backend systems.

To emphasize the principles learned in this course, this project navigates through various AWS services and tools. It outlines using the AWS Systems Manager for parameter storing, ensuring secure and efficient application configuration management. Additionally, the project explores the realms of high availability and automatic scaling, ensuring that the organization's website remains resilient and responsive, even under fluctuating traffic conditions.

The report is structured to provide a holistic view of the project. It begins with a summary of the problem and the client's requirements, followed by a detailed account of the chosen design approach that aligns with AWS's well-architected framework. A meticulous cost analysis of the proposed solution ensures that the project remains economically viable, while diagrams and screenshots visually represent the implementation process. The narrative also includes a candid discussion of the challenges encountered during the project and the strategies to overcome them.

In conclusion, the report summarizes the tasks accomplished and recommendations for future improvements. An appendix section acknowledges the contributions of each team member, highlighting the collaborative effort that underpins this project. This document is a testament to the technical competencies acquired and a reflection of the team's ability to apply theoretical knowledge to real-world scenarios, marking a significant milestone in our academic and professional journey.

**Proposal Outline**

1. **Client Needs Analysis**

**1.1 Background and Current Challenges**

The Example Social Research Organization, a fictitious yet representative non-profit entity, has experienced significant growth in its digital outreach. This growth, however, has not been without challenges:

* Performance Issues: As the website’s popularity surged, users began to experience decreased responsiveness. The underlying cause is the limited capacity of the current hosting solution to handle the increased traffic.
* Security Concerns: Though unsuccessful, an attempted ransomware attack highlights vulnerabilities in the website’s security posture. This incident underscores the need for a more secure hosting environment to safeguard sensitive data and maintain user trust.

**1.2 Detailed Analysis of Current Website and Database Implementation**

The current implementation of the organization’s website and database presents several areas of concern:

* EC2 Instance and Public Subnet Usage:
  + Observation: The website hosting is on a single Amazon EC2 instance within a public subnet. This setup exposes the website and database to potential security risks and does not leverage the full scalability and reliability features offered by AWS.
  + Implications: The public subnet positioning increases vulnerability to external threats, and the single-instance approach leads to potential single points of failure, impacting the website’s availability and performance.
* MySQL Database Deployment:
  + Observation: The MySQL database resides on the same EC2 instances as the website. While this configuration simplifies management, it also intertwines the database’s performance and security with the web servers.
  + Implications: Hosting the database on the same instance as the web server can lead to resource contention, where both the database and the web application compete for the same computational resources. This setup can significantly impact performance, especially during peak traffic periods.

**1.3 Initial Conclusions**

The current architecture, while functional, needs to align with best practices for high availability, scalability, and security, which are essential for the growing needs of the Example Social Research Organization. The next section of this report will outline a proposed solution design that addresses these challenges and sets a foundation for a robust, secure, and scalable web application infrastructure on AWS. This design will optimize the website’s performance, enhance its security posture, and ensure its scalability to accommodate future growth and demand.

1. **Solution Design**

**2.1 AWS Architecture Overview**

This architecture leverages AWS services to ensure scalability, reliability, and security:

* Selection of AWS Services and Justifications:
  + Amazon EC2: Hosts the PHP application, providing flexibility and control over the server.
  + Amazon RDS: Manages the MySQL database, enhancing performance, scalability, and separation from the application server.
  + Elastic Load Balancer: Distributes incoming traffic across multiple EC2 instances, ensuring high availability and fault tolerance.
  + Amazon VPC: Enhances network security, allowing the creation of a private subnet for the database.
  + AWS Auto Scaling: Automatically adjusts the number of EC2 instances, scaling the application in response to demand.
  + AWS Systems Manager Parameter Store: Secures and manages application configurations and secrets.
* Proposed Architecture Diagram:
  + The diagram shows the interconnectivity of these services, with the application server in a public subnet, the database in a private subnet, and the utilization of ELB and Auto Scaling.

**2.2 Deployment Strategy**

The strategy focuses on efficient setup, security, and performance optimization:

* PHP Application Deployment on EC2:
  + Sets up a t2.small EC2 instance for the PHP application
  + Configures the instance with necessary security groups and IAM roles to restrict access.
  + Deploys the application and tests its connectivity with external services.
* MySQL Database Implementation:
  + Migrates the database to Amazon RDS to utilize its managed services, including automated backups, patch management, and scalability.
  + Establishes a private subnet within the VPC for the RDS instance to enhance security.

**2.3 Security Measures**

Key measures protect the infrastructure and data:

* Network Security (VPC, Security Groups, etc.):
  + Implements a VPC with public and private subnets.
  + Creates security groups for EC2 and RDS with minimal access rules.
  + Uses Network Access Control Lists (NACLs) for additional security.
* IAM Roles and Policies:
  + Defines IAM roles for EC2 instances for secure interactions with other AWS services.
  + Establishes strict IAM policies following the principle of least privilege.
* Data Encryption and Backup Strategies:
  + Enables encryption for data at rest and in transit.
  + Sets up regular backups for the RDS instance.
  1. **Scalability and High Availability**

Ensures the website handles varying loads efficiently:

* Load Balancing and Auto-Scaling:
  + Configures ELB to distribute traffic evenly across multiple EC2 instances.
  + Sets up Auto Scaling to add or remove instances based on predefined metrics like CPU utilization.
* Multi-AZ Deployment for RDS:
  + Implements a Multi-AZ deployment for the RDS instance to ensure high availability and automatic failover.
  1. **Systems Manager Parameter Store**

Manages application configurations securely:

* Secure Storage of Database Credentials:
  + Stores database connection strings, usernames, and passwords in the Parameter Store.
  + Ensures these parameters are encrypted, and access is tightly controlled.
* Automation and Management of Application Configurations:
  + Automates the retrieval of configuration parameters by the application.
  + Regularly updates and audits the parameters for security and compliance.

1. **Implementation and Testing**

**3.1 Deployment Steps**

The deployment of the solution involves a series of carefully executed steps to ensure a secure and functional environment:

* EC2 Instance Setup and Configuration:
  + Launches a t2.small EC2 instance for the PHP application.
  + Installs and configures the necessary software and applications on the EC2 instance.
  + Applies appropriate security groups and IAM roles to the instance for secure access.
* Database Migration and Setup:
  + Migrates the existing MySQL database to Amazon RDS, ensuring data integrity and minimal downtime.
  + Configures the RDS instance in a private subnet within the Amazon VPC for enhanced security.
* Load Balancer and Auto-Scaling Configuration:
  + Sets up Elastic Load Balancing to manage incoming traffic efficiently.
  + Configures AWS Auto Scaling to maintain optimal performance under varying load conditions.

**3.2 Testing and Validation**

After deployment, comprehensive testing validates the functionality, performance, and security of the solution:

* Performance Testing:
  + Conducts load testing to evaluate the application’s response under high-traffic conditions.
  + Monitors resource utilization to ensure efficient scaling and load balancing.
* Security Vulnerability Assessments:
  + Performs security audits to identify potential vulnerabilities within the infrastructure.
  + Implements necessary updates and patches in response to the findings.
* Functional Testing:
  + Verifies that all components of the application function as intended.
  + Ensures seamless interaction between the PHP application and the MySQL database.

**3.3 Screenshots and Descriptions of Key Implementation Steps**

Documenting the implementation process is crucial for transparency and future reference:

* Deployment Documentation:
  + Captures screenshots of the AWS console during critical steps of the deployment process.
  + Provides detailed descriptions alongside each screenshot to explain the actions taken.
* Testing Logs and Reports:
  + Maintains logs of all testing activities, including performance and security tests.
  + Summarizes the outcomes of these tests in a concise report format.

1. **Cost Analysis**

**4.1 Breakdown of AWS Service Costs**

This section provides a detailed cost analysis for the AWS services used in the project:

* Amazon EC2 Costs:
  + Calculating the costs associated with running the t2.small EC2 instance, including any associated storage and network usage.
  + Evaluating the pricing for different instance types and reserved instances for potential cost savings.
* Amazon RDS Costs:
  + Analyzing the pricing for the RDS instance, including storage, I/O requests, and backup storage costs.
  + Considering the implications of Multi-AZ deployment on costs.
* Elastic Load Balancing and Auto Scaling Costs:
  + Assessing the costs associated with using Elastic Load Balancing, including data processing and additional instance costs for Auto Scaling.
* AWS Systems Manager Parameter Store Costs:
  + Examining any costs related to the usage of the Parameter Store, focusing on parameter storage and retrieval transactions.

**4.2 Cost Optimization Strategies**

This part explores various strategies to optimize AWS costs while maintaining performance and security:

* Selecting Appropriate Resource Sizes:
  + Choosing the suitable instance sizes and types to balance performance needs with cost-effectiveness.
  + Regularly reviewing and adjusting resources based on usage patterns.
* Utilizing Reserved Instances and Savings Plans:
  + Considering AWS Reserved Instances and Savings Plans for long-term cost reductions.
  + Analyzing usage patterns to determine the feasibility of these options.
* Implementing Efficient Scaling Policies:
  + Designing Auto Scaling policies that effectively manage resource utilization, avoiding over-provisioning.
* Monitoring and Cost Management Tools:
  + Utilizing AWS tools like CloudWatch and Cost Explorer to monitor and manage costs.
* Setting up alerts for cost thresholds to maintain budget control.

**4.3 Comparison with Previous Hosting Solution**

This section compares the costs of the new AWS-based solution with the previous hosting arrangement:

* Cost Comparison Analysis:
  + Providing a comparative analysis of the overall costs, including hosting, maintenance, and scalability.
  + Highlighting the benefits and potential savings of the AWS solution.
* Performance and Security Value:
  + Discussing how the AWS solution adds value in terms of improved performance, scalability, and security, which may justify any additional costs.

1. **Challenges and Solutions**

**5.1 Overview of Issues Encountered**

In this section, the report highlights the various challenges faced during the implementation of the project and how they were addressed:

* Performance Optimization:
  + Issue: Initial performance tests indicated slower response times under high load.
  + Solution: Adjusted Auto Scaling policies and optimized database queries to improve response times.
* Security Hardening:
  + Issue: Potential vulnerabilities identified during the security audit.
  + Solution: Applied security patches, tightened IAM policies, and implemented additional network access controls.
* Database Migration Complexities:
  + Issue: Data integrity issues and downtime during the migration process.
  + Solution: Used AWS Database Migration Service to streamline the migration and minimize downtime.

**5.2 Problem-Solving Approaches**

This part of the report outlines the systematic approaches taken to solve the problems:

* Collaborative Troubleshooting:
  + Approach: Worked as a team to diagnose issues, leveraging diverse skills and perspectives.
  + Outcome: Enhanced problem-solving efficiency and team learning.
* Utilization of AWS Support and Documentation:
  + Approach: Consulted AWS documentation and support resources for best practices and troubleshooting tips.
  + Outcome: Gained deeper insights into AWS services and their optimal usage.
* Iterative Testing and Feedback:
  + Approach: Adopted an iterative approach to testing and improvements, incorporating feedback at each stage.
  + Outcome: Gradually refined the system for better performance and security.

**5.3 Lessons Learned**

This section reflects on the valuable insights gained from the challenges:

* Importance of Thorough Planning:
  + Lesson: Emphasized the need for detailed planning in cloud architecture and deployment.
* Adaptability and Continuous Learning:
  + Lesson: Highlighted the importance of being adaptable and continuously updating skills, especially in the rapidly evolving field of cloud computing.
* Security as a Continuous Process:
  + Lesson: Reinforced the concept that security is not a one-time task but a continuous process requiring regular updates and vigilance.

1. **Conclusions and Future Considerations**

The project successfully deployed a PHP application on AWS EC2 and migrated the MySQL database to Amazon RDS, significantly enhancing the Example Social Research Organization's website performance and security. Key achievements include:

1. Efficient and Scalable Architecture: Implementation of Elastic Load Balancing and Auto Scaling.
2. Robust Security: Strengthened security through network configurations, IAM roles, and data encryption.
3. Cost-Effectiveness: Balancing performance improvements with cost-effective AWS service choices.
4. Project's Impact on the Client

The new AWS infrastructure directly benefits the client by:

1. Handling Increased Traffic: Ensuring the website remains responsive and efficient under high loads.
2. Securing Sensitive Data: Reducing the risk of security breaches, crucial for the organization's credibility.
3. Future Growth Support: Providing a scalable and adaptable platform for future growth.
4. Recommendations for Future Enhancements

Looking ahead, the following enhancements are recommended:

1. Advanced Monitoring: Incorporate AWS monitoring tools for proactive management.
2. Adopt Additional AWS Services: Consider services like AWS Lambda for serverless operations and Amazon S3 for data storage.
3. Regular Security Reviews: Maintain a robust security posture through ongoing assessments and updates.
4. User Experience Focus: Continuously optimize the user interface and experience based on user feedback.

**Appendixes**

**Appendix A: Statement of Contributions**

**Member 1: Brandon Luff:**

Discussion Contribution: Our group convened outside of regular class hours for project discussions, maintaining communication through text and email. This collaborative effort allowed us to align our ideas and approaches effectively.

Lab Contribution: Focused on building the inner workings of the cloud-based solution to fulfill the requirements as necessary in order to provide the company with their ideal final product. Unfortunately, we found it difficult to get the site to actually query the mySQL database, however, the database did contain the proper files as tested in the lab environment. Additionally, my lab will be the one we are submitting for the project.

Report Contribution: Provided the figures and screenshots necessary to display and connect our work in the lab to this paper.

**Member 2: Benjamin Gaudlip**

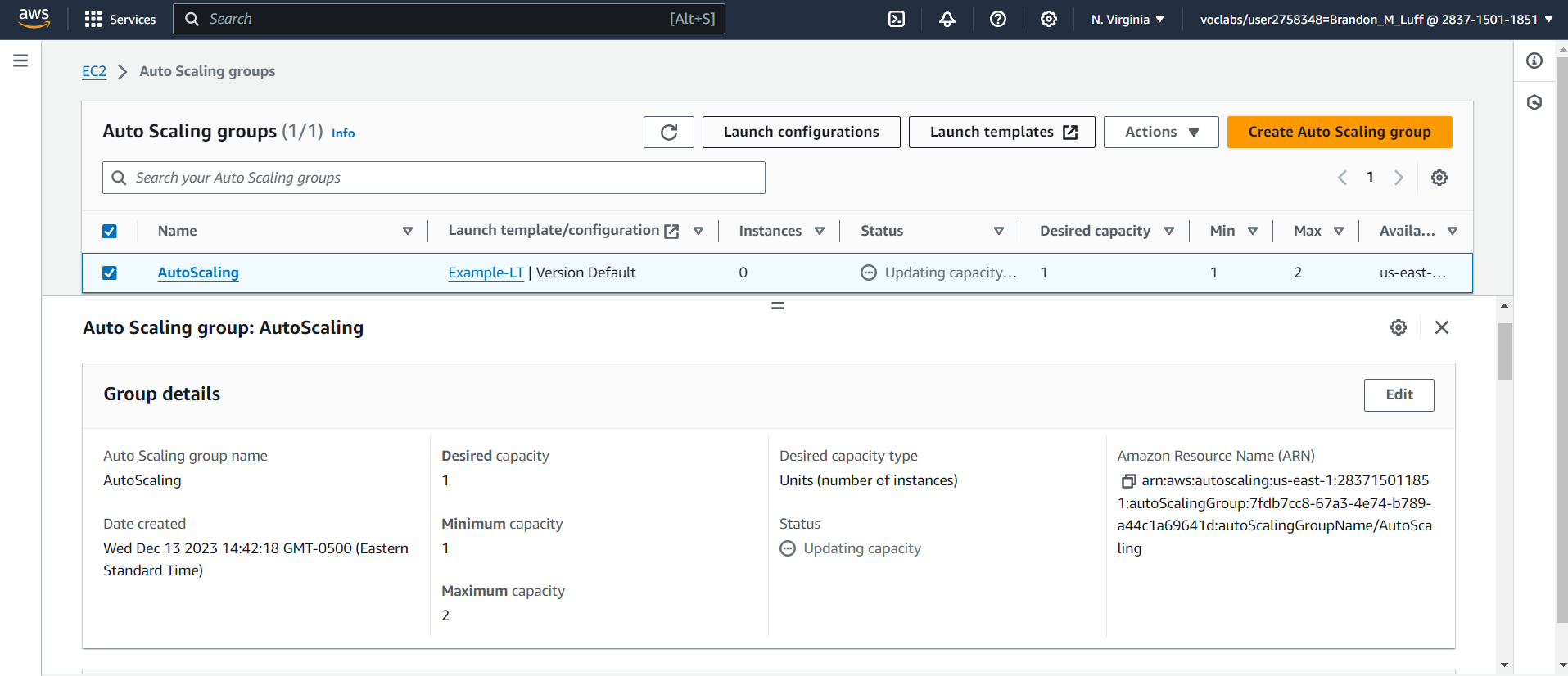
Discussion Contribution: Our group convened outside of regular class hours for project discussions, maintaining communication through text and email. This collaborative effort allowed us to align our ideas and approaches effectively.

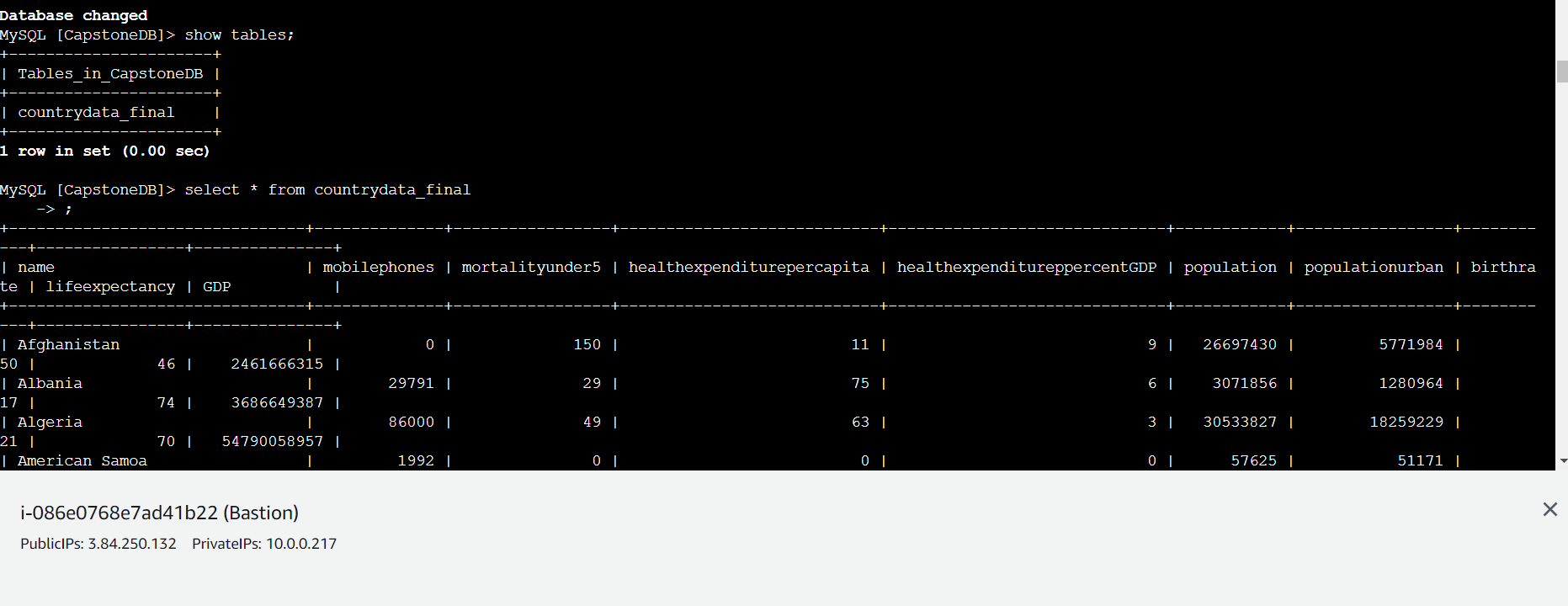
Lab Contribution: Provided lab support and consultation. Offered guidance and technical advice during the implementation phase, ensuring adherence to best practices and resolving complex technical challenges.

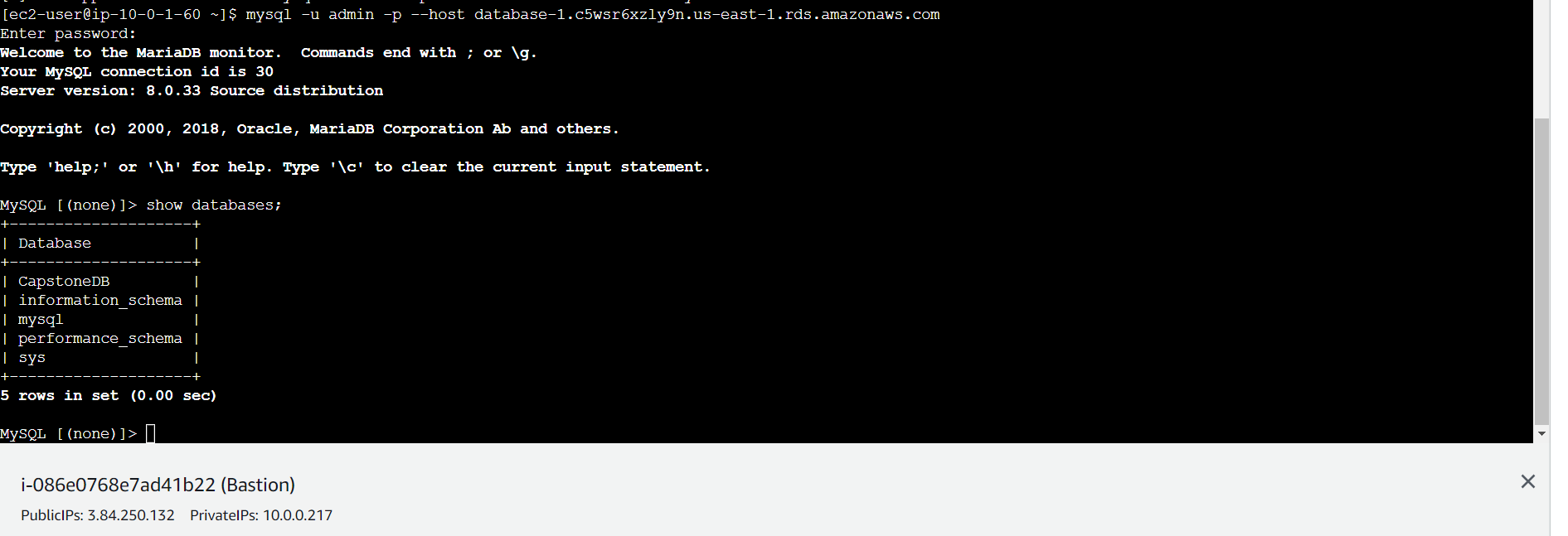
Report Contribution: Authored the project report, excluding figures. Drafted a

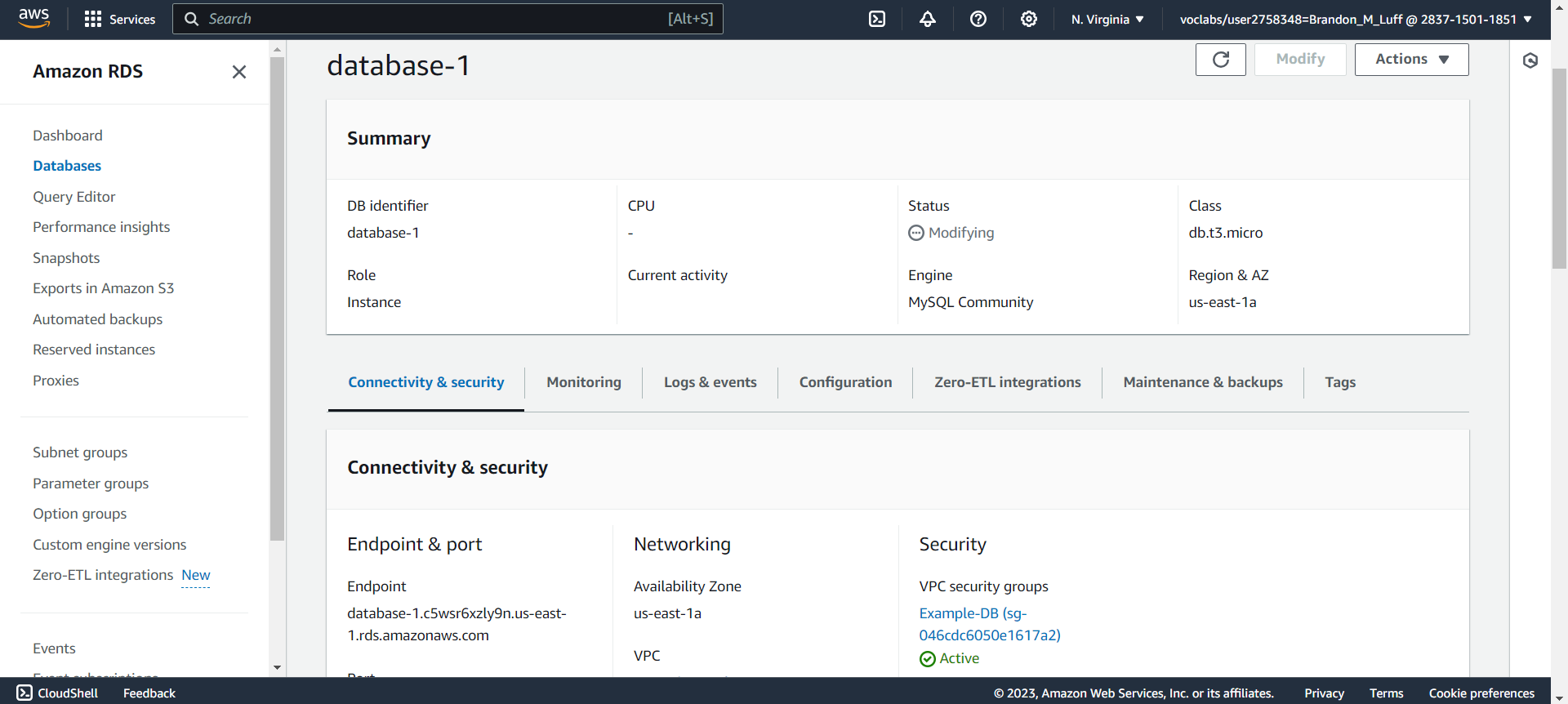
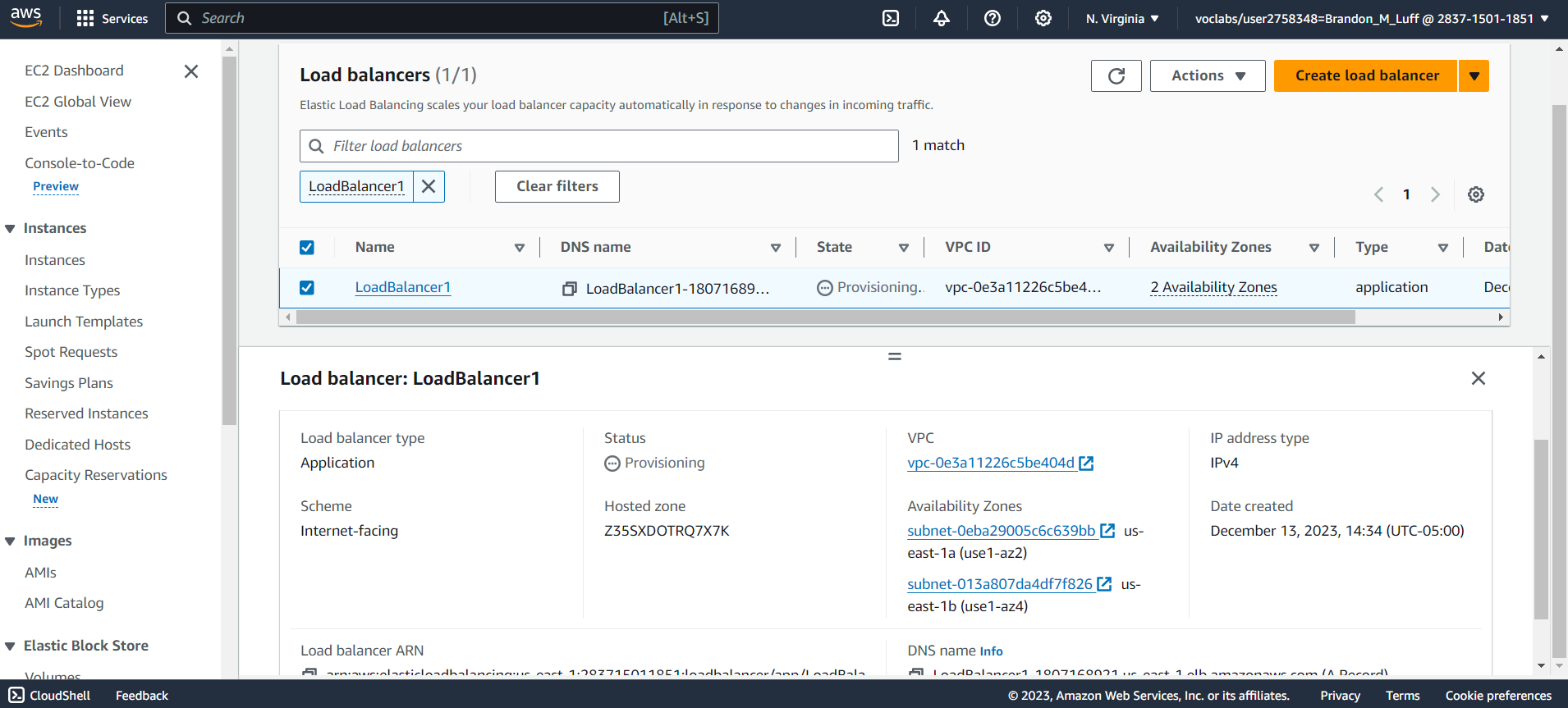
comprehensive document covering all aspects of the project, including the introduction, solution design, implementation and testing, cost analysis, challenges and solutions, and conclusion with future recommendations.

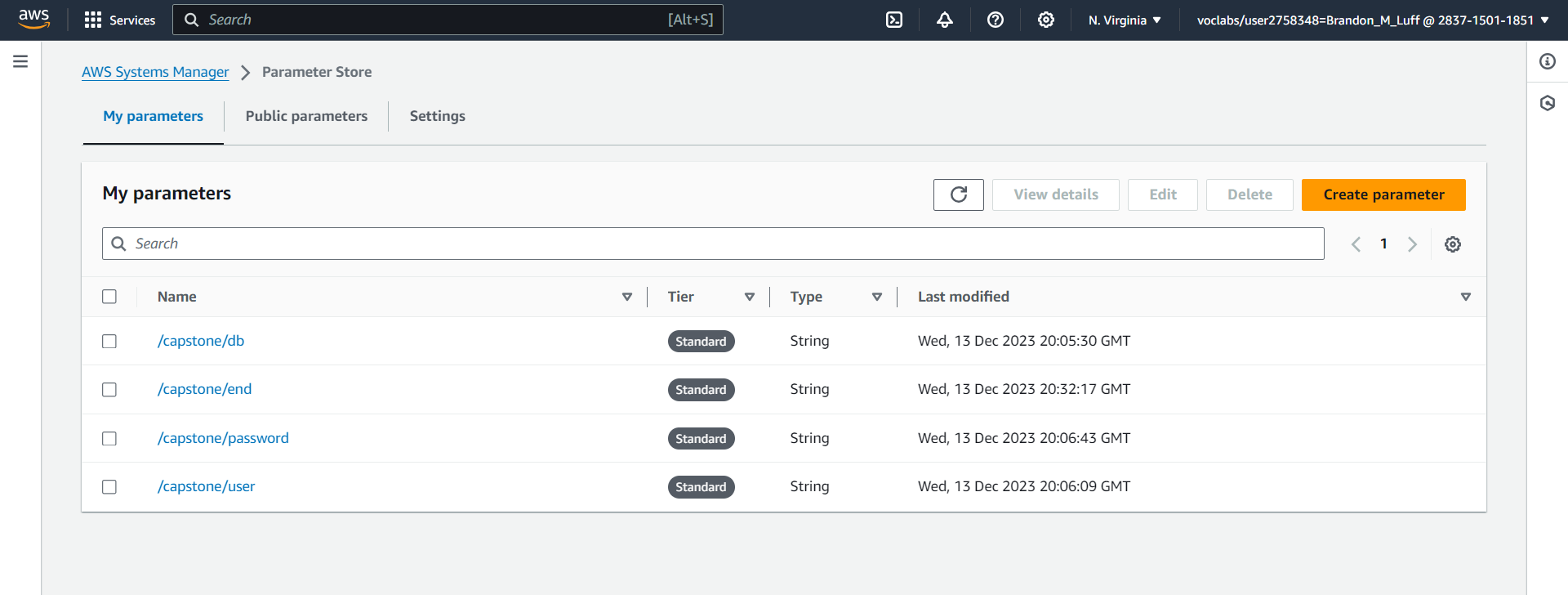
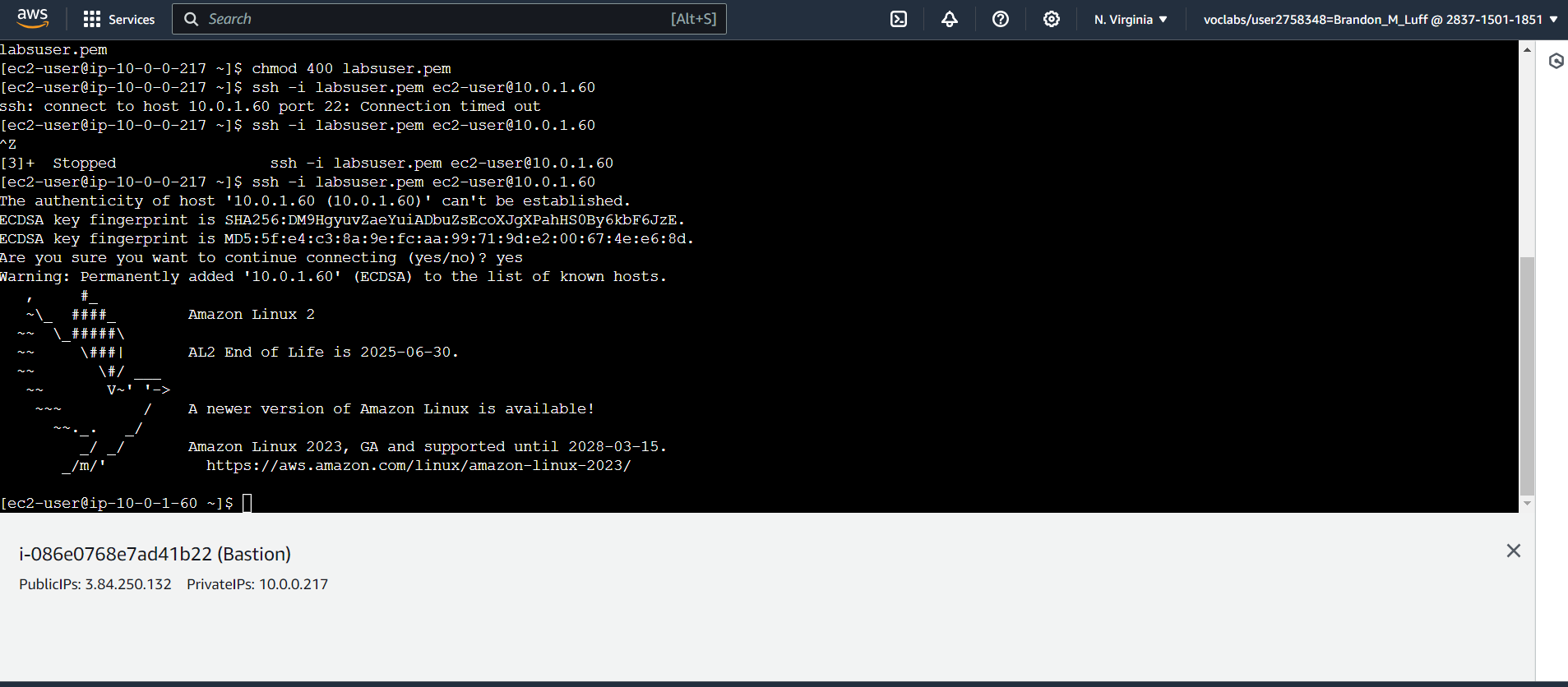
**Figures**

*Autoscaling group creation*****

*Database after adding assets*****

*Database prior to adding assets*****

*RDS creation******Load Balancer Creation*****

*Parameter creation to allow access from Bastion Host******Successful SSH login to EC2 instance and database*****

**References**

FARRAR, STRAUS & GIROUX. (2024). *Amazon Web Services*. Amazon. https://aws.amazon.com/console/