Project Report: Building a Cloud-Based File Sharing Application on Google Cloud Platform

Performed by: Aisha Ait

Table of Contents

---

1. Executive Summary

Provide a brief overview of the project's goals, the technologies utilized, and the outcomes achieved:

Project Overview:
This project aimed to create a cloud-based file-sharing app using Google Cloud Platform (GCP). The app is designed to let users easily upload, share, and manage their files securely and efficiently. I focused on building a reliable cloud environment that ensures the app performs well and keeps user data safe.

Technologies Used:
I used several tools from Google Cloud, including Google Compute Engine for running virtual machines, Google Cloud Storage for storing files, Google Cloud VPC for managing the network, and Google Cloud IAM for handling user permissions and security. These tools all work together to create a solid cloud solution.

Outcomes Achieved:
The project resulted in a fully functional file-sharing app that's both scalable and secure. Users can manage their files easily while enjoying peace of mind knowing their data is protected. Overall, this project highlights how useful cloud services can be for developing modern applications.

2. Introduction

Introduce the concept of cloud computing and its significance in modern application development. Discuss the motivations behind selecting Google Cloud Platform for the project:

Cloud computing is a way to store and access data and applications over the internet instead of using local computers or servers. It's important for modern app development because it offers flexibility, scalability, and cost savings. Developers can quickly launch applications without worrying about hardware limits, which helps them innovate faster and work better together.

In today's fast-paced world, cloud computing allows developers to create, test, and scale applications more easily. It cuts infrastructure costs, improves reliability, and enhances security. Plus, cloud services make updates and maintenance simpler, so businesses can adapt quickly to changing needs.

I picked Google Cloud Platform for my project because:

1. It is obligatory for this midterm 😊
2. GCP can handle different workloads easily, which is perfect for my file-sharing app.
3. It offers everything I need, from computing to storage, allowing for seamless integration.
4. GCP keeps user data safe with features like Identity and Access Management (IAM).
5. Its data centers around the world ensure fast service for users everywhere.
6. GCP regularly updates its services, giving me access to the latest technology.

By choosing Google Cloud, I could build a secure and efficient file-sharing application that meets my users' needs.

3. Project Objectives

List the specific objectives of the project, such as building a secure and scalable file sharing application, and ensuring high availability:

The main objectives of the project are to build a secure file-sharing application that protects user data through encryption and strong access controls, ensuring that only authorized users can access sensitive information. Additionally, the goal is to create a scalable system that can handle many users and files simultaneously, allowing the application to grow as user demand increases without compromising performance.
Ensuring high availability is also crucial, so users can access the app anytime without interruptions, minimizing downtime through reliable infrastructure and redundancy. The project aims to implement easy-to-use features for uploading and managing files, making it simple for users to share, organize, and retrieve their documents quickly. Lastly, optimizing performance is key to providing a smooth user experience, including fast loading times and efficient file transfer capabilities, so users can seamlessly interact with the application.

4. Cloud Computing Overview

Describe the principles of cloud computing, including its key benefits and deployment models (IaaS, PaaS, SaaS):

Key benefits of cloud computing include:

- **Cost Efficiency**: Reduces the need for physical hardware and maintenance, allowing businesses to pay only for the resources they use.
- **Scalability**: Users can easily scale resources up or down based on demand, making it suitable for varying workloads.
- **Accessibility**: Cloud services can be accessed from anywhere with an internet connection, enhancing collaboration and flexibility.
- **Disaster Recovery**: Offers robust backup and recovery solutions to protect data and ensure business continuity.
- **Automatic Updates**: Service providers manage software updates and security patches, reducing the burden on IT teams.

Cloud computing is typically categorized into three main deployment models:

Infrastructure as a Service (IaaS)	Platform as a Service (PaaS)	Software as a Service (SaaS)
Provides virtualized computing resources over the internet. Users can rent servers, storage, and networking capabilities without needing to manage physical hardware. Examples include Google Compute Engine and Amazon EC2.	Offers a platform that allows developers to build, deploy, and manage applications without dealing with the underlying infrastructure. PaaS solutions provide tools and services for application development, testing, and deployment. Examples include Google App Engine and Microsoft Azure.	Delivers software applications over the internet on a subscription basis. Users can access software without needing to install or maintain it on local devices. Examples include Google Workspace and Salesforce.

These models provide flexibility and choice, allowing businesses to select the right services based on their specific needs and goals.

5. **Google Cloud Platform: Core Services**

- ***Core Services Overview****: Describe the main services provided by GCP that are relevant to the project.*

Google Cloud Platform (GCP) offers a wide range of core services that are essential for building and managing applications in the cloud. These services include:

- **Compute Services:** These provide virtual machines and container orchestration for running applications. They allow users to deploy, manage, and scale applications efficiently.
- **Storage Services:** GCP offers various storage solutions for different types of data, including object storage, file storage, and database storage, enabling users to store and manage their data securely and cost-effectively.
- **Networking Services:** These include Virtual Private Cloud (VPC), load balancing, and Content Delivery Network (CDN) capabilities, which help in managing network traffic and ensuring secure connectivity.
- **Database Services:** GCP provides managed database solutions, both relational (like Cloud SQL) and NoSQL (like Firestore), for storing and querying data effectively.
- **Identity and Security Services**: Tools like Identity and Access Management (IAM) help in managing user permissions and securing applications against unauthorized access.

- ***Service Selection****: Outline the services chosen for the application and their specific roles.*

For this file-sharing application, I selected the following GCP services:

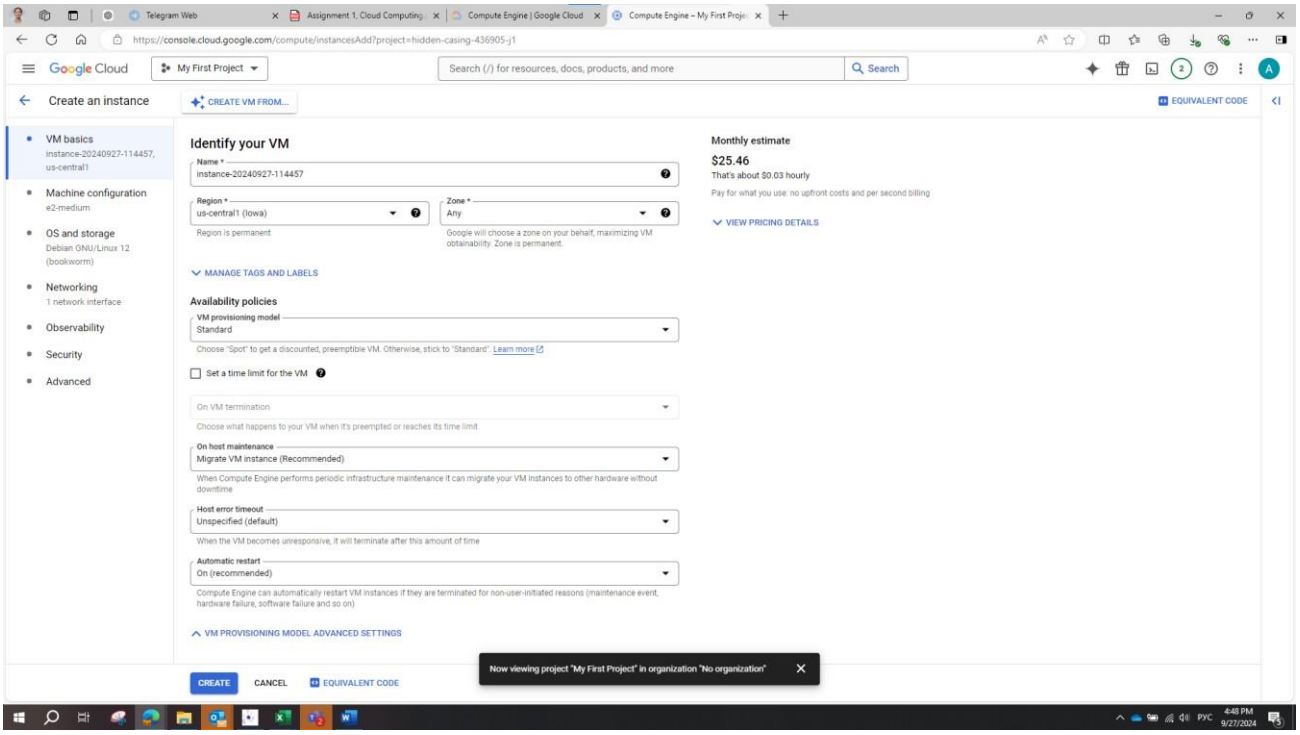
- **Google Compute Engine:** This service is used to set up and run virtual machines that host the backend of the application. It provides the necessary computing resources to handle user requests and file processing.
- **Google Cloud Storage:** This service is chosen for file storage. It enables users to upload, download, and manage files securely and efficiently, with the capability to scale as storage needs grow.
- **Google Cloud VPC:** VPC is used to manage networking for the application, allowing for secure communication between different components, such as the VM and the storage service.
- **Google Cloud IAM:** This service is implemented to control user access and permissions, ensuring that only authorized users can interact with the application and access sensitive data.

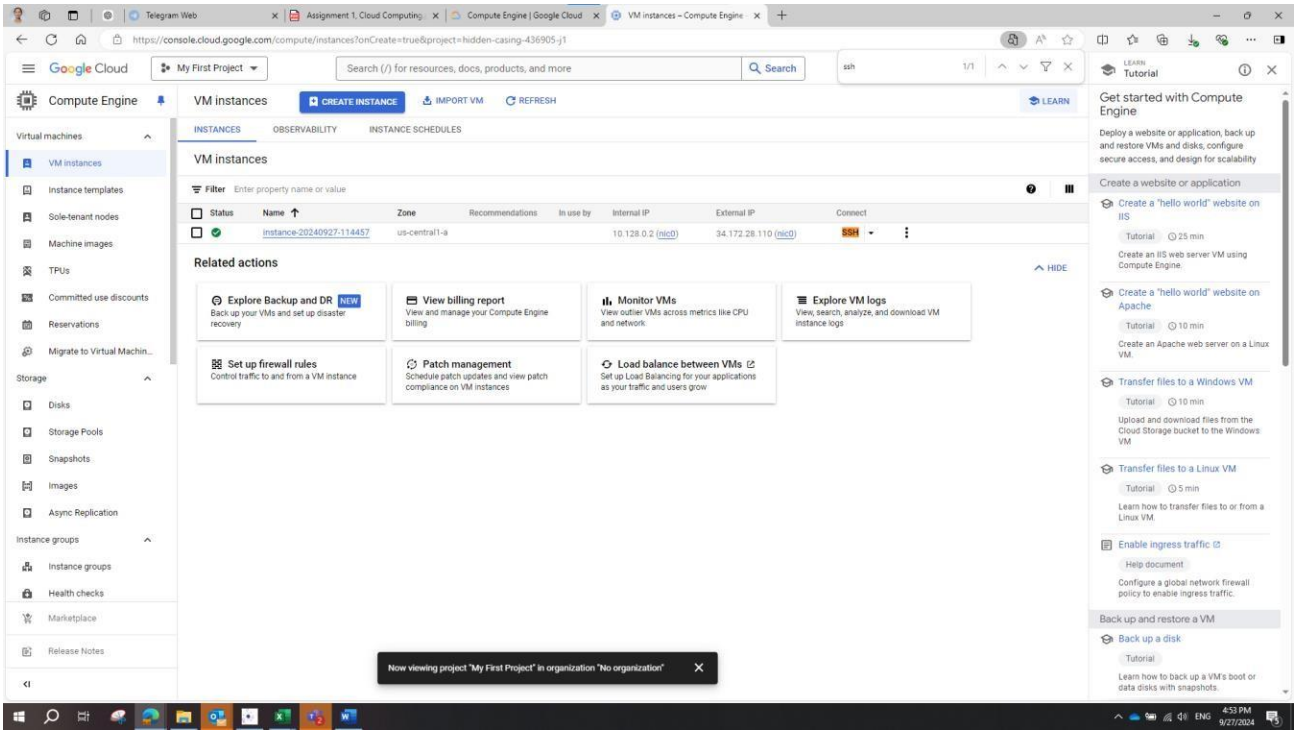
By selecting these services, I was able to create a secure, scalable, and efficient file-sharing application that meets user needs and complies with best practices for cloud architecture.

6. **Virtual Machines in Google Cloud**

VM Setup*: Detail the steps taken to set up and configure virtual machines using Google Compute Engine.*

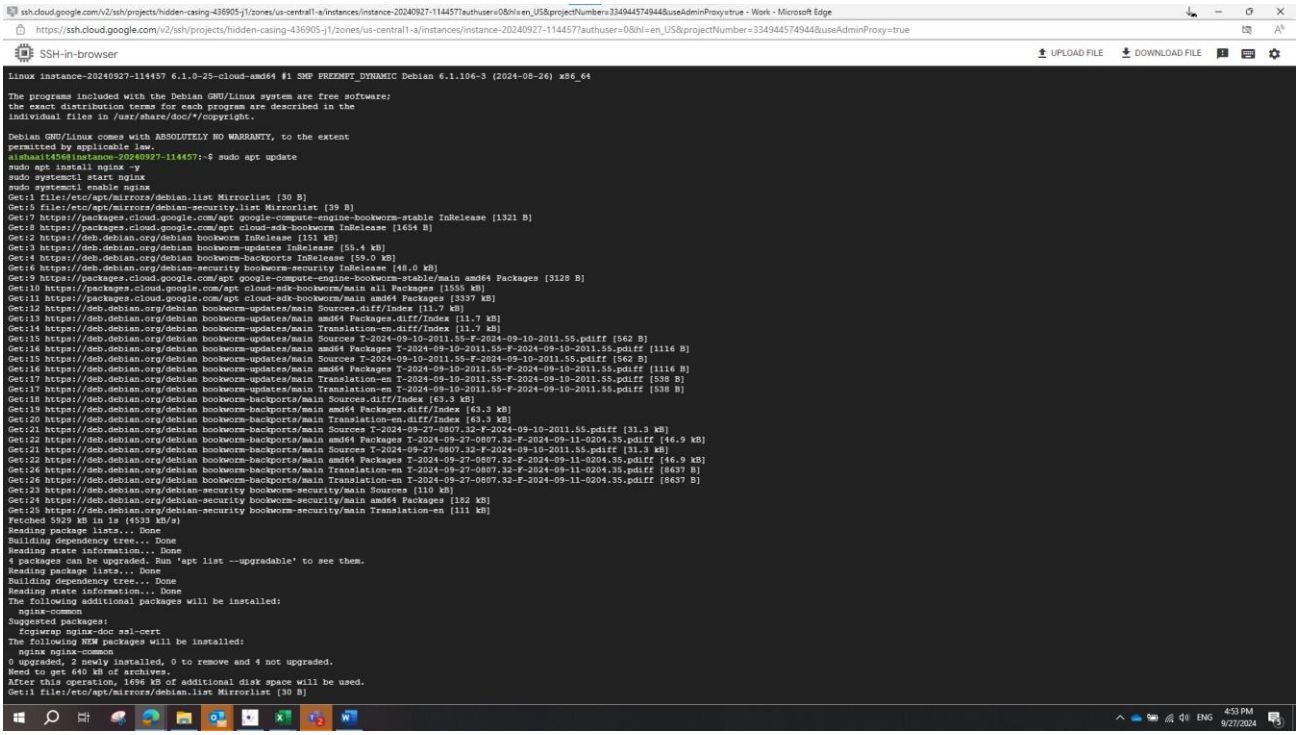
Step 1 – I navigated to the Compute Engine section in the Google Cloud Console and selected VM instances. I clicked on "Create Instance" and configured the machine type and region based on my project requirements. I also selected an appropriate operating system, ensuring it met the application needs.

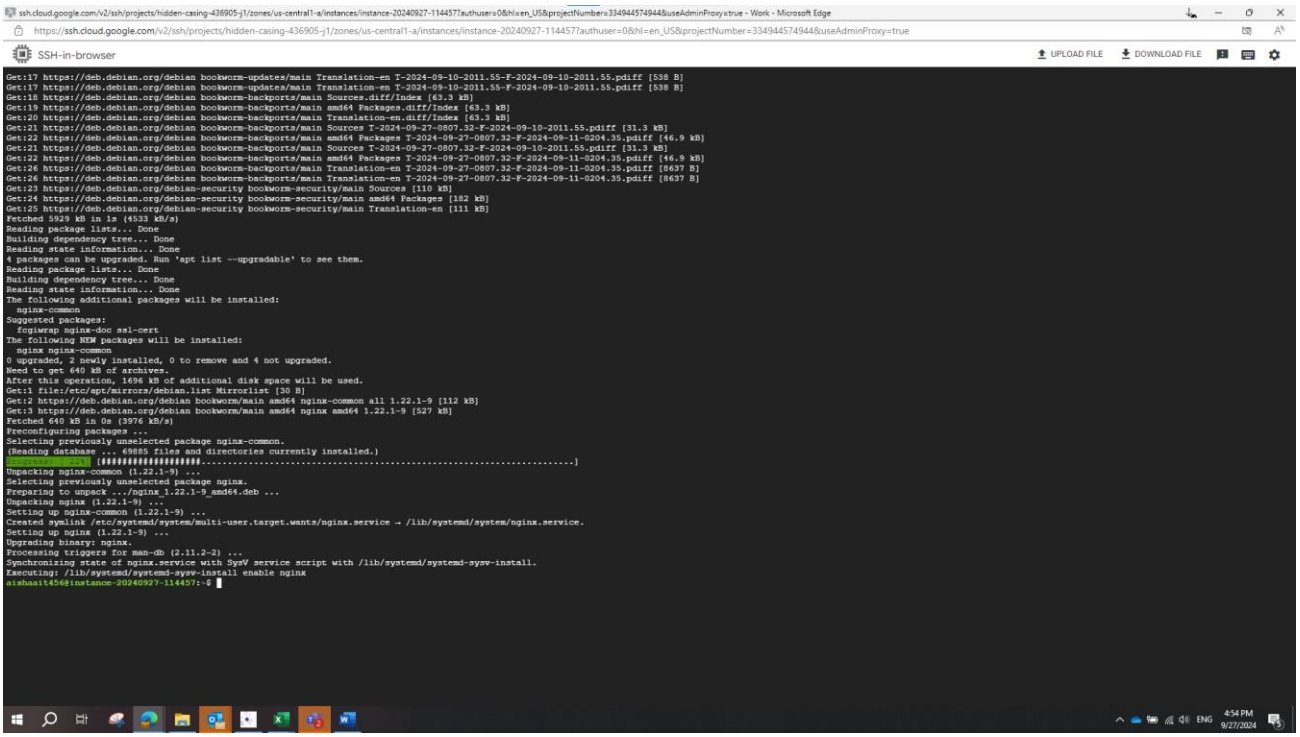




Step 2 – After the VM was created, I connected to it via SSH directly from the Google Cloud Console. Once connected, I installed Nginx, which is a web server, using the following commands:

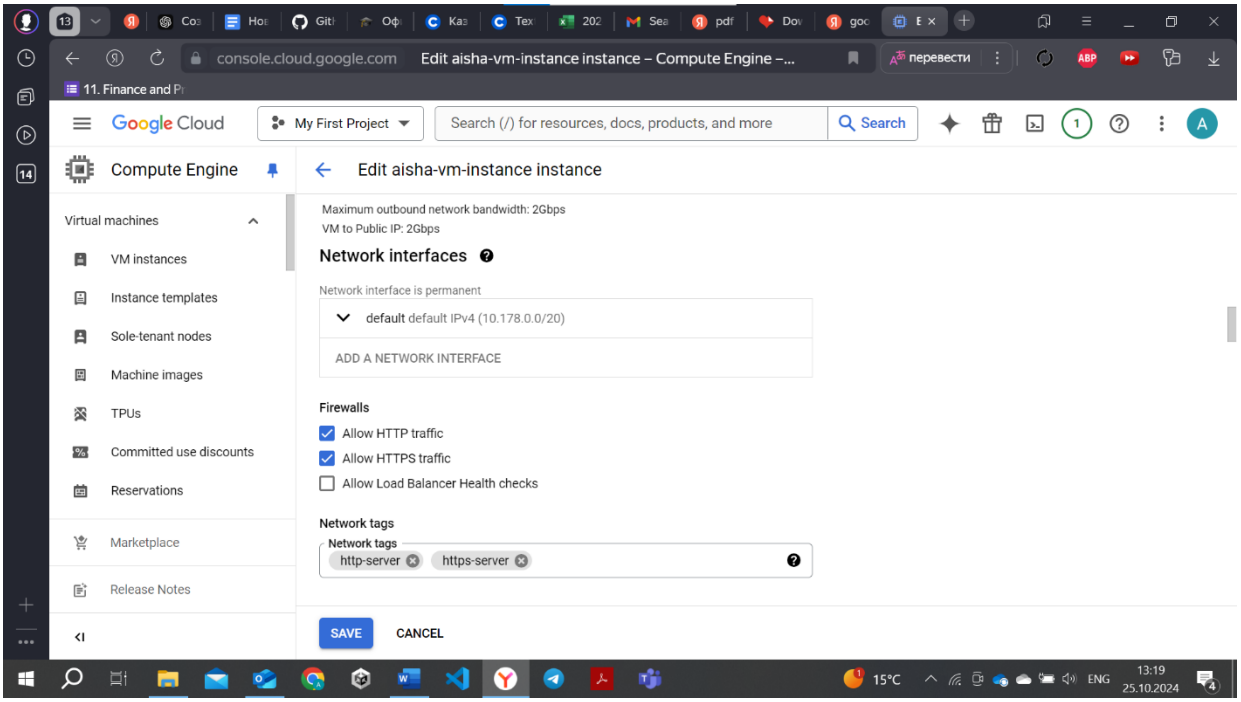
- I updated the package list: `sudo apt update`
- I installed Nginx: `sudo apt install nginx`
- After installation, I started the Nginx service: `sudo systemctl start nginx`

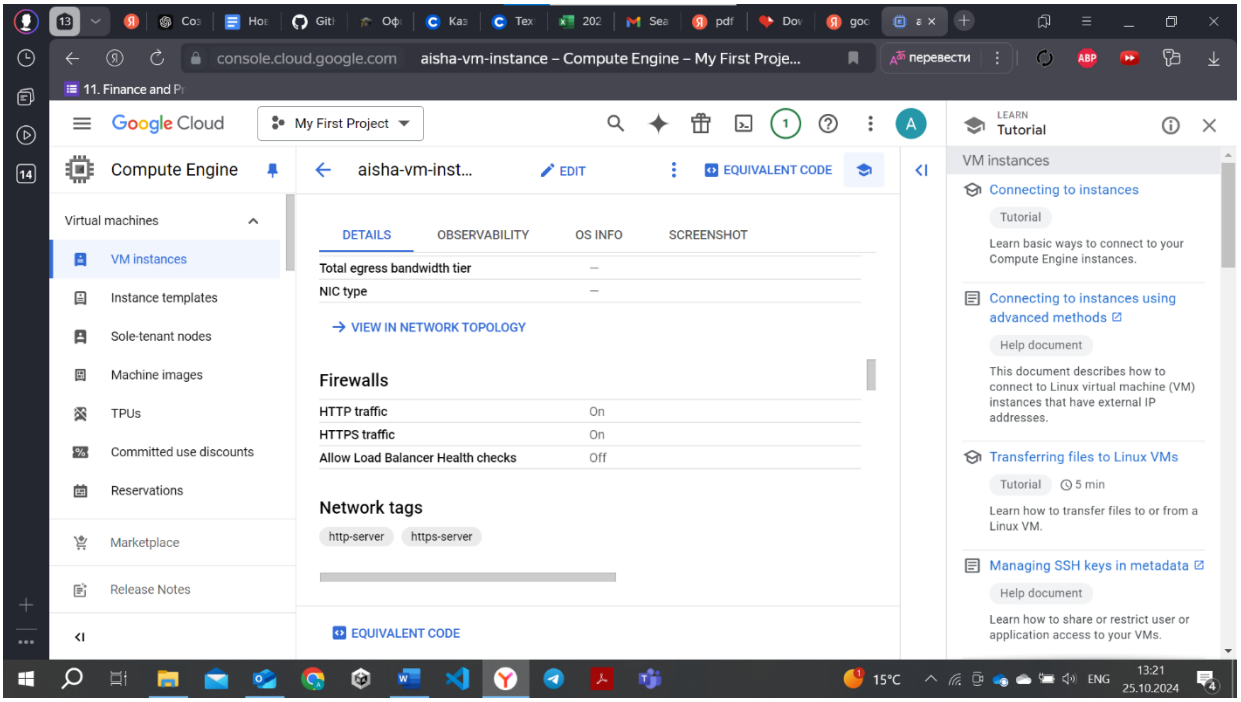




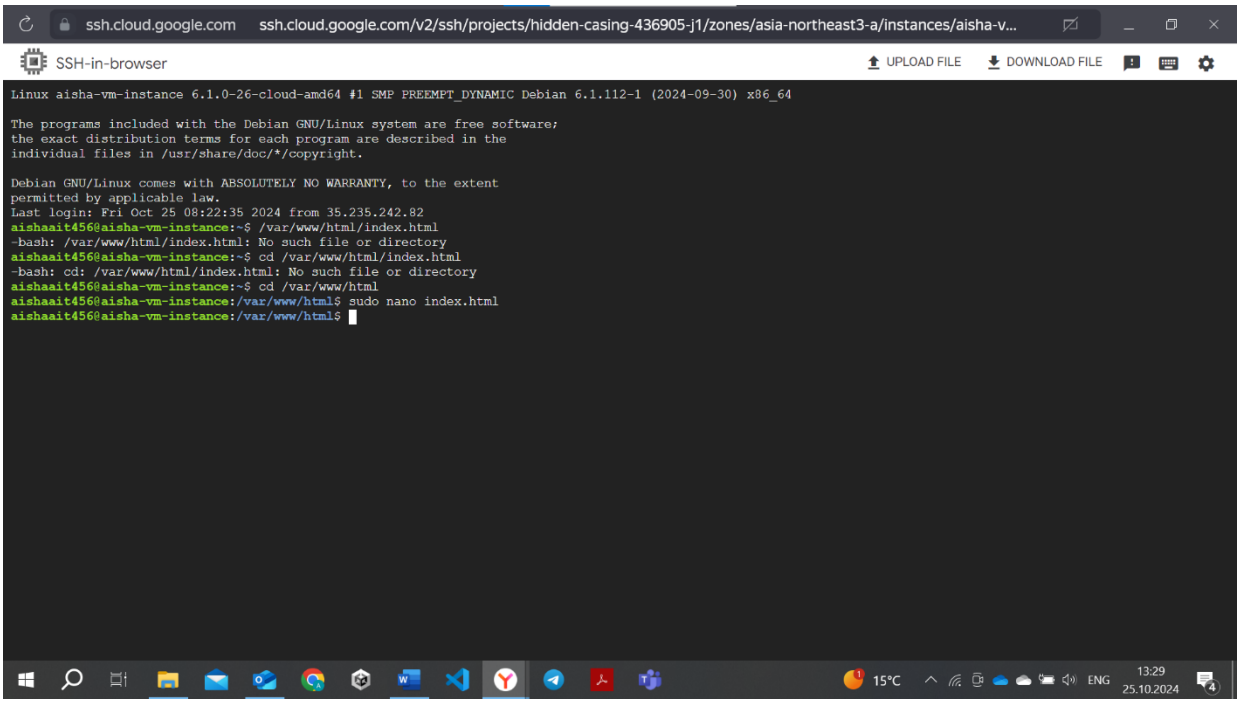
- ***Application Deployment****: Discuss the deployment of the application's backend on the VM, including configuration details.*

The backend of the file-sharing application was deployed on the VM. I configured Nginx to serve as the web server for handling incoming HTTP requests. This involved setting up the server block to define how the application responds to different URLs and routes.

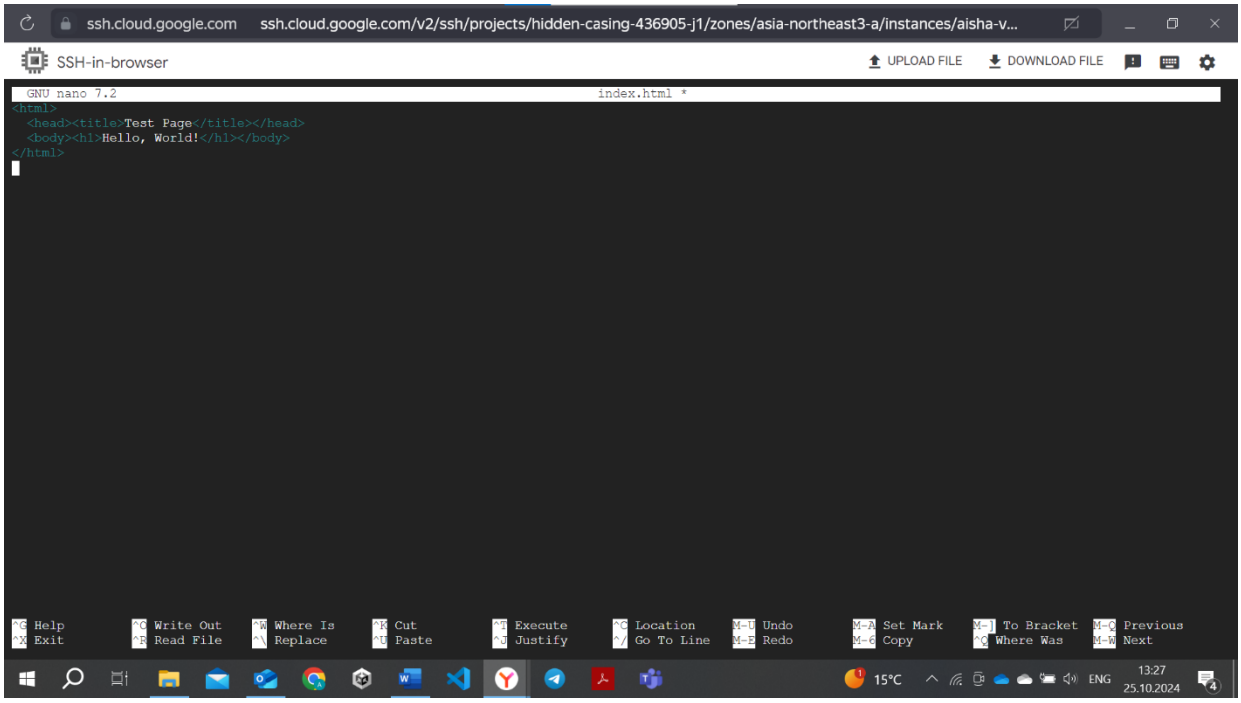




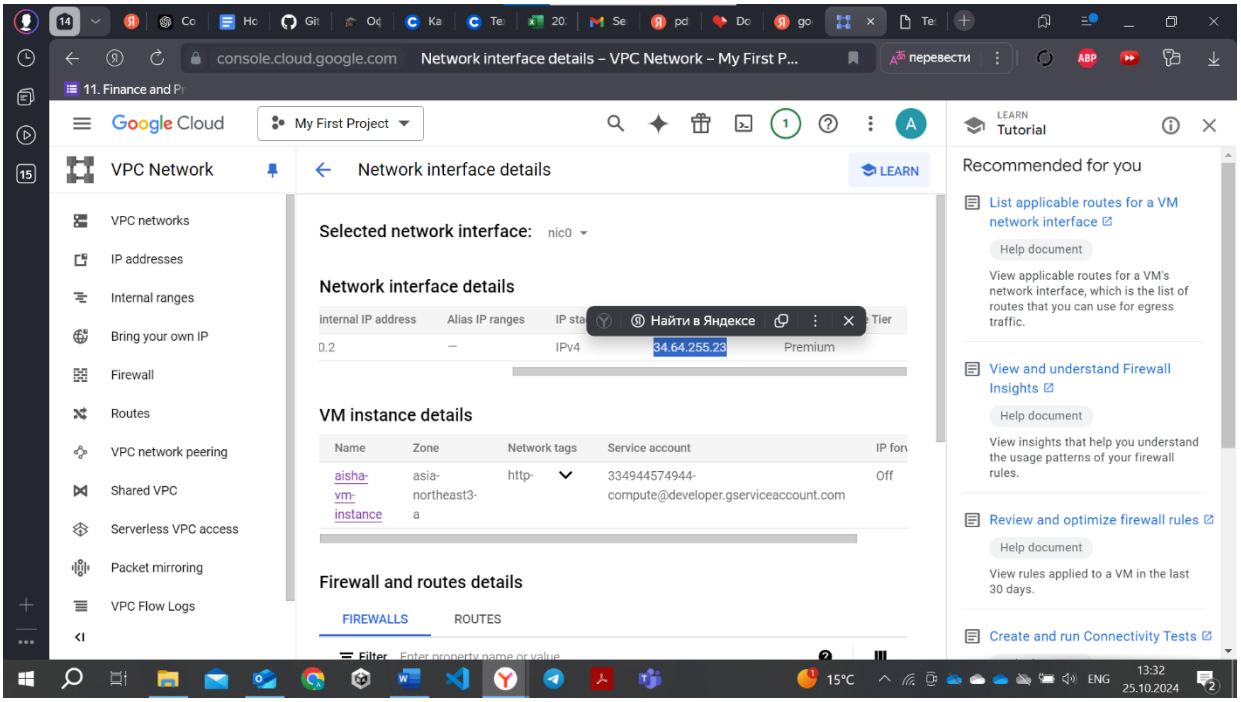
I uploaded the application files to the VM's web directory, typically located at /var/www/html/. I also ensured proper permissions were set for these files to allow Nginx to serve them correctly. To test the deployment, I accessed the VM's public IP address in a web browser, confirming that the application was running smoothly and the backend was responding as expected.

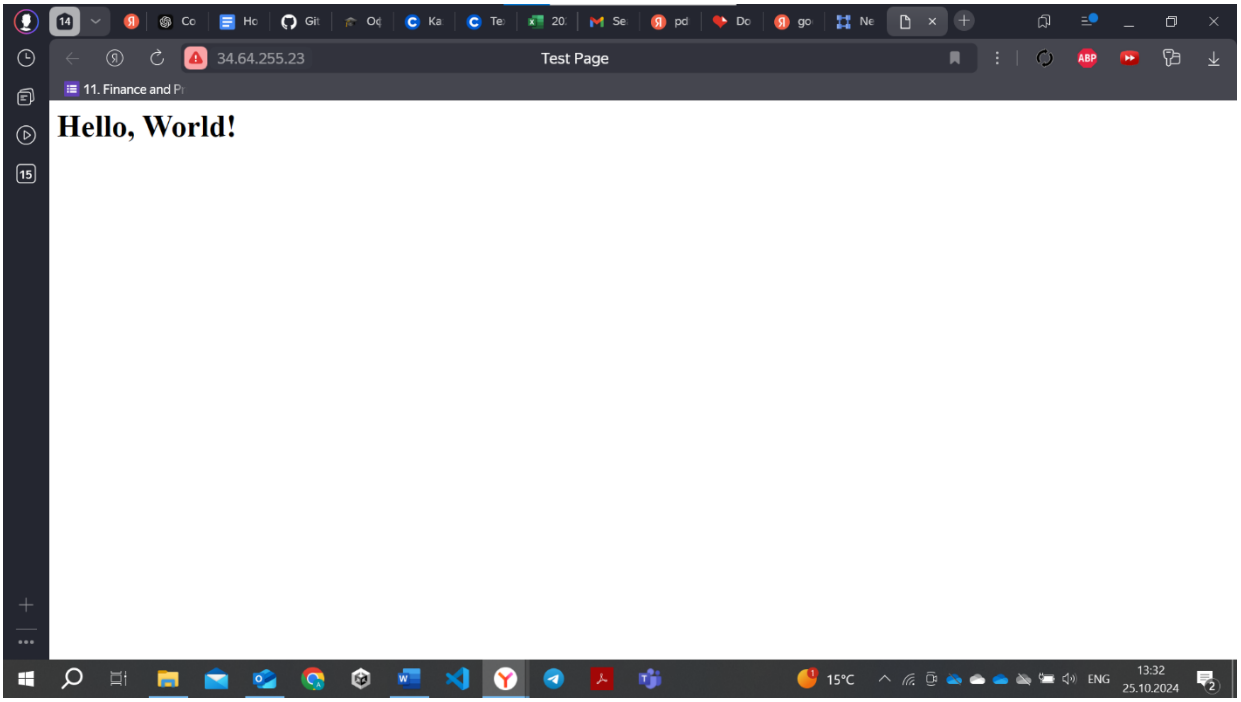


- Use a text editor like nano or vim to create an HTML file: `sudo nano index.html`



- Opened the VM's public IP address in a browser to confirm the web server is running, and the test page is displayed correctly.

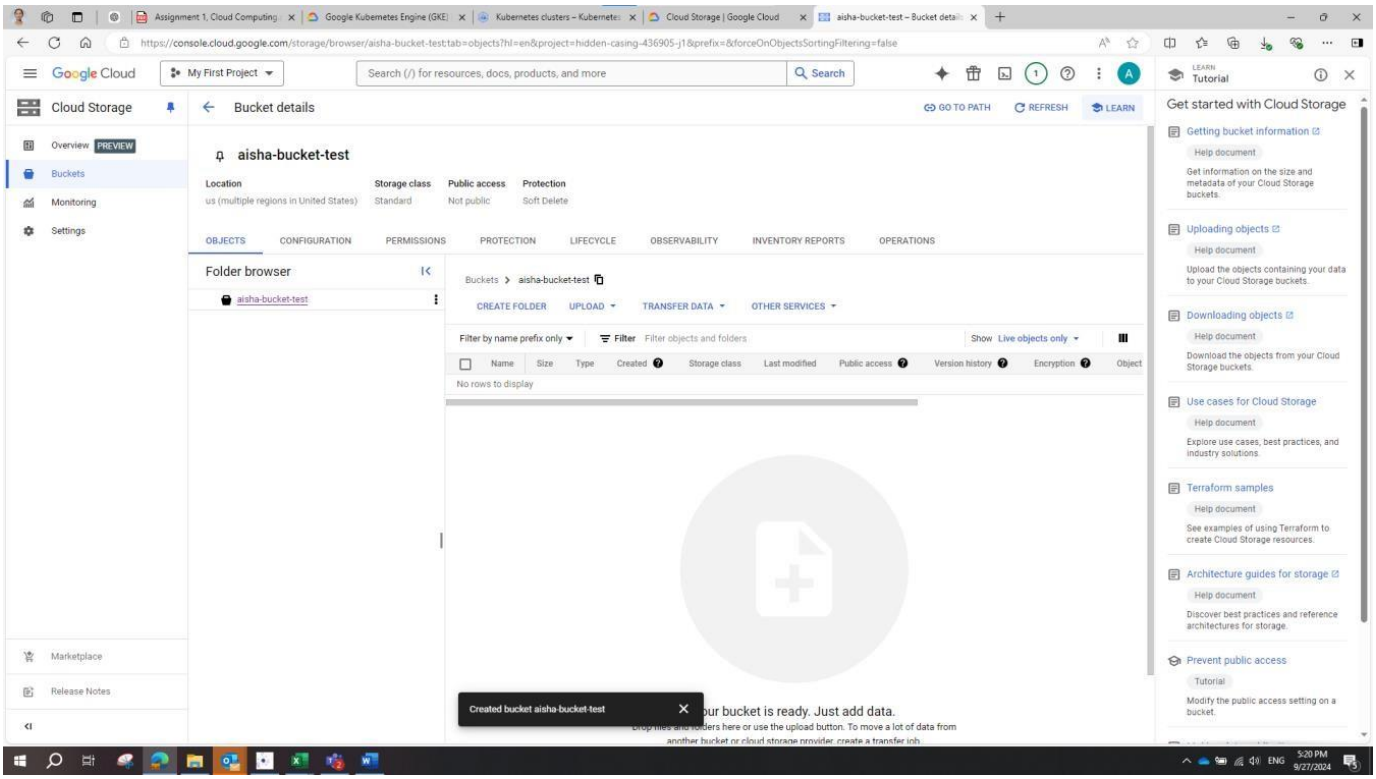


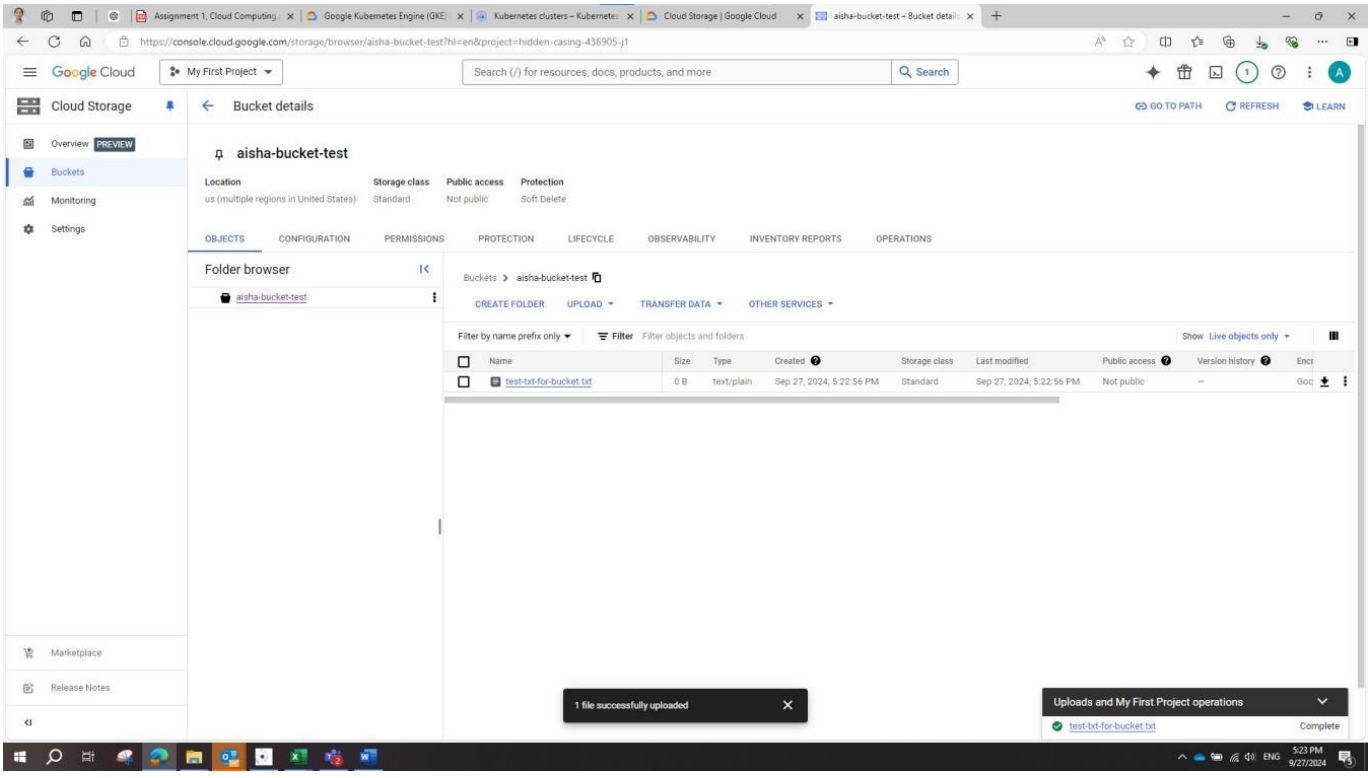


7. **Storage Solutions in Google Cloud**

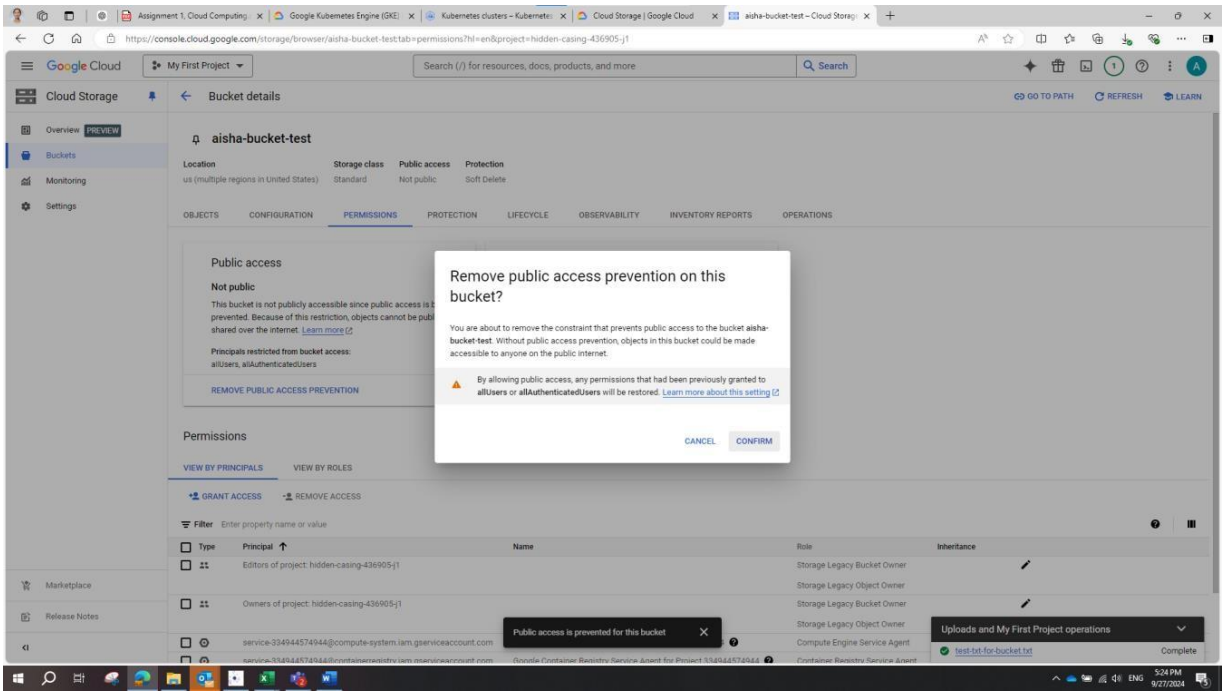
- ***Cloud Storage Implementation****: Explain how Google Cloud Storage was utilized for file management.*

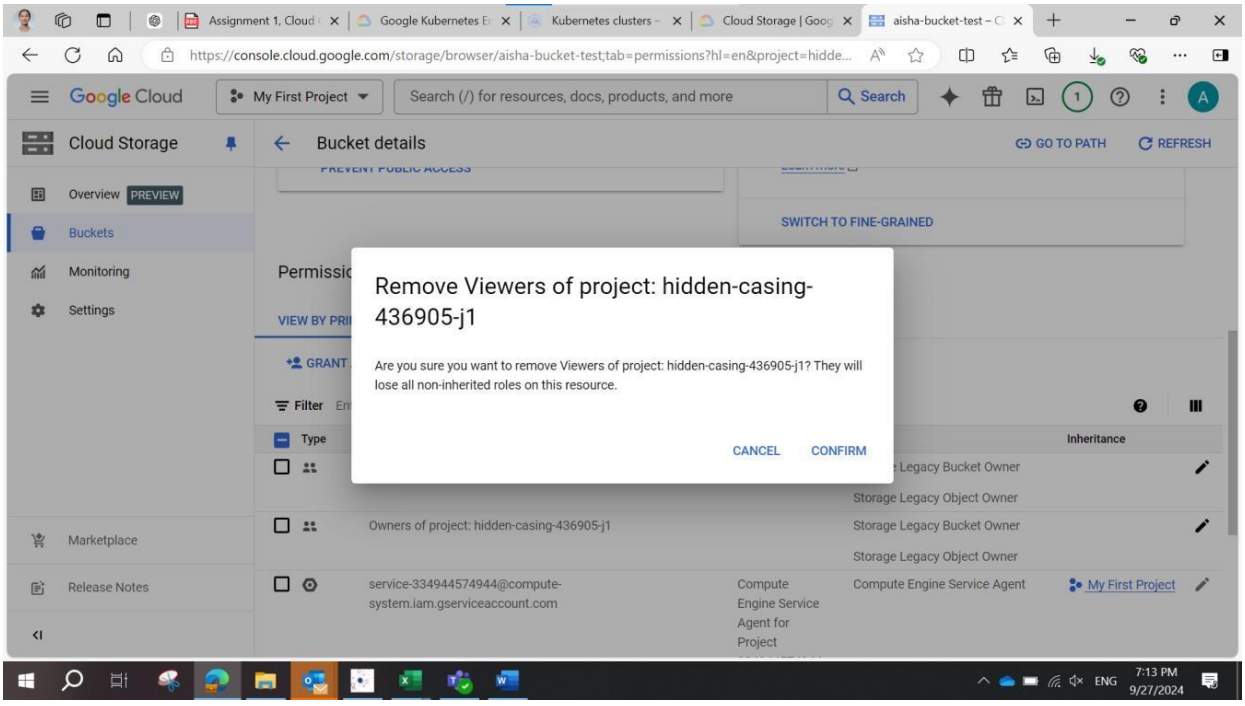
Google Cloud Storage was used to handle all file management for the application, providing scalable and secure storage for uploaded files. I created a Cloud Storage bucket specifically for this purpose, enabling users to store, access, and manage files through a central location.

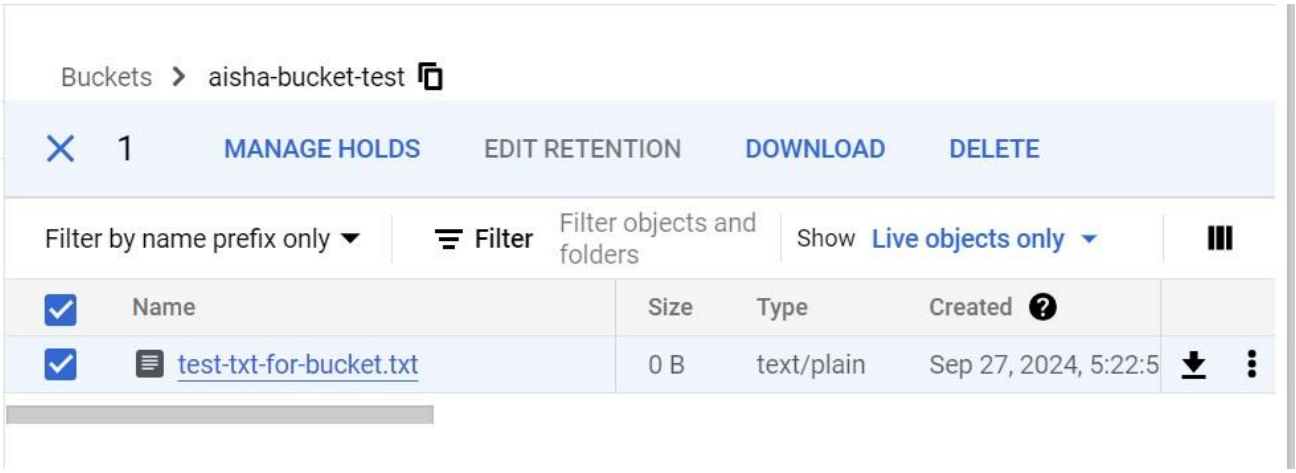

Google Cloud
My First Project
Search (/) for resources, docs, products, and more
Search
Cloud Storage
Bucket details
GO TO PATH
REFRESH
LEARN
Overview
PREVIEW
Buckets
Monitoring
Settings
aisha-bucket-test
Location
us (multiple regions in United States)
Storage class
Standard
Public access
Not public
Protection
Soft Delete
OBJECTS
CONFIGURATION
PERMISSIONS
PROTECTION
LIFECYCLE
OBSERVABILITY
INVENTORY REPORTS
OPERATIONS
Folder browser
aisha-bucket-test
Buckets > aisha-bucket-test
CREATE FOLDER
UPLOAD
TRANSFER DATA
OTHER SERVICES
Filter by name prefix only
Filter
Filter objects and folders
Show
Live objects only
Name
Size
Type
Created
Storage class
Last modified
Public access
Version history
Encryption
Object
No rows to display
Created bucket aisha-bucket-test
ur bucket is ready. Just add data.
Drop files and folders here or use the upload button. To move a lot of data from another bucket or cloud storage provider, create a transfer job.
Marketplace
Release Notes
Tutorial
Get started with Cloud Storage
Getting bucket information
Help document
Get information on the size and metadata of your Cloud Storage buckets.
Uploading objects
Upload the objects containing your data to your Cloud Storage buckets.
Downloading objects
Download the objects from your Cloud Storage buckets.
Use cases for Cloud Storage
Explore use cases, best practices, and industry solutions.
Terraform samples
See examples of using Terraform to create Cloud Storage resources.
Architecture guides for storage
Discover best practices and reference architectures for storage.
Prevent public access
Modify the public access setting on a bucket.
ENG
5:20 PM
9/27/2024


Google Cloud
My First Project
Search (/) for resources, docs, products, and more
Search
Cloud Storage
Bucket details
GO TO PATH
REFRESH
LEARN
Overview
PREVIEW
Buckets
Monitoring
Settings
aisha-bucket-test
Location
us (multiple regions in United States)
Storage class
Standard
Public access
Not public
Protection
Soft Delete
OBJECTS
CONFIGURATION
PERMISSIONS
PROTECTION
LIFECYCLE
OBSERVABILITY
INVENTORY REPORTS
OPERATIONS
Folder browser
aisha-bucket-test
Buckets > aisha-bucket-test
CREATE FOLDER
UPLOAD
TRANSFER DATA
OTHER SERVICES
Filter by name prefix only
Filter
Filter objects and folders
Show
Live objects only
Name
Size
Type
Created
Storage class
Last modified
Public access
Version history
test-txt-for-bucket.txt
0 B
text/plain
Sep 27, 2024, 5:22:56 PM
Standard
Sep 27, 2024, 5:22:56 PM
Not public
1 file successfully uploaded
Uploads and My First Project operations
test-txt-for-bucket.txt
Complete
Marketplace
Release Notes
ENG
5:23 PM
9/27/2024

The bucket's permissions were configured to ensure that only authorized users could upload or download files. To further secure file access, I used Google Cloud's Identity and Access Management (IAM) to set permissions based on user roles.
With Fine-grained access, you can manage individual file permissions by going to the file's settings in the Permissions tab and adding/removing specific IAM roles or setting public/private access for that specific file. You can also apply access policies using signed URLs for temporary access.



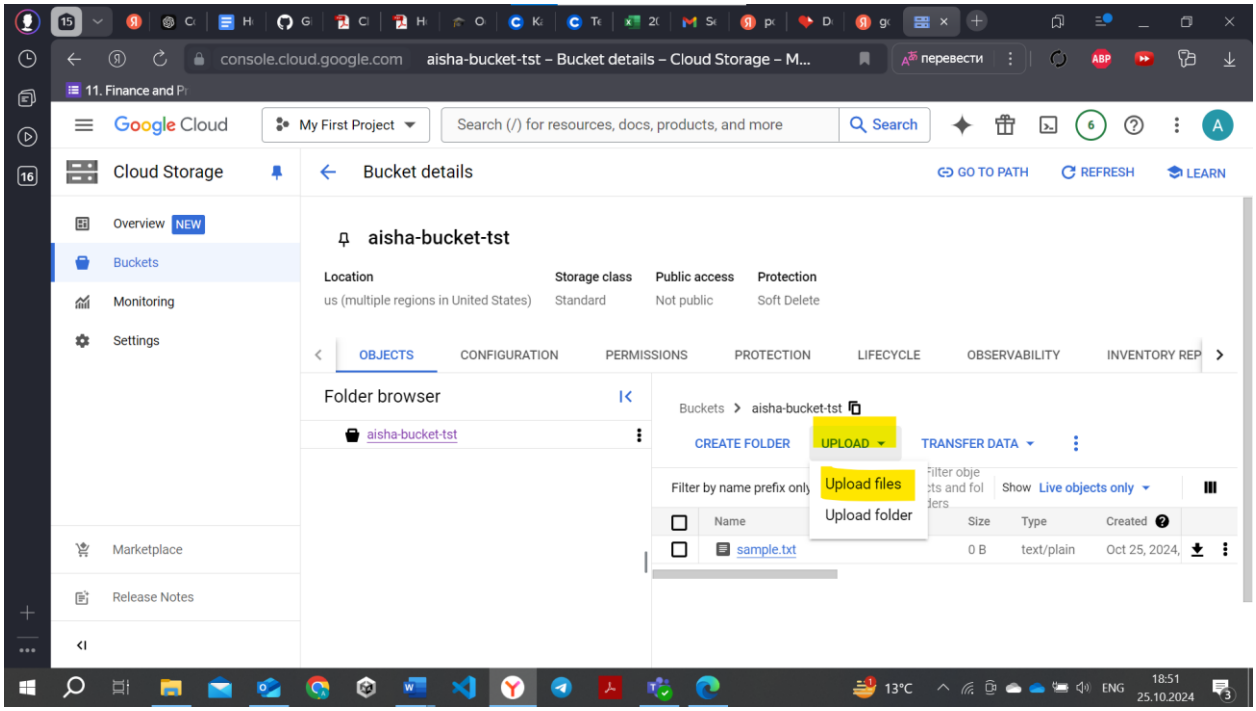




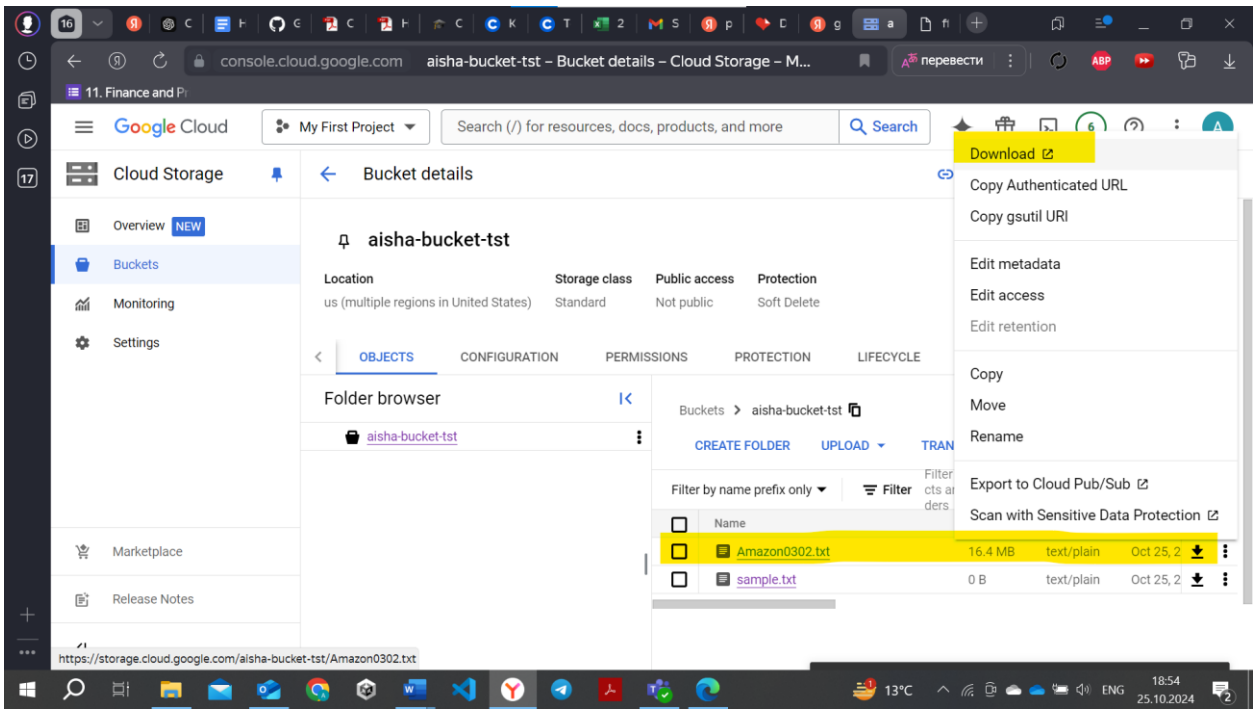
- ***File Management System****: Describe the system for uploading, downloading, and managing files, including any relevant code snippets or configurations.*

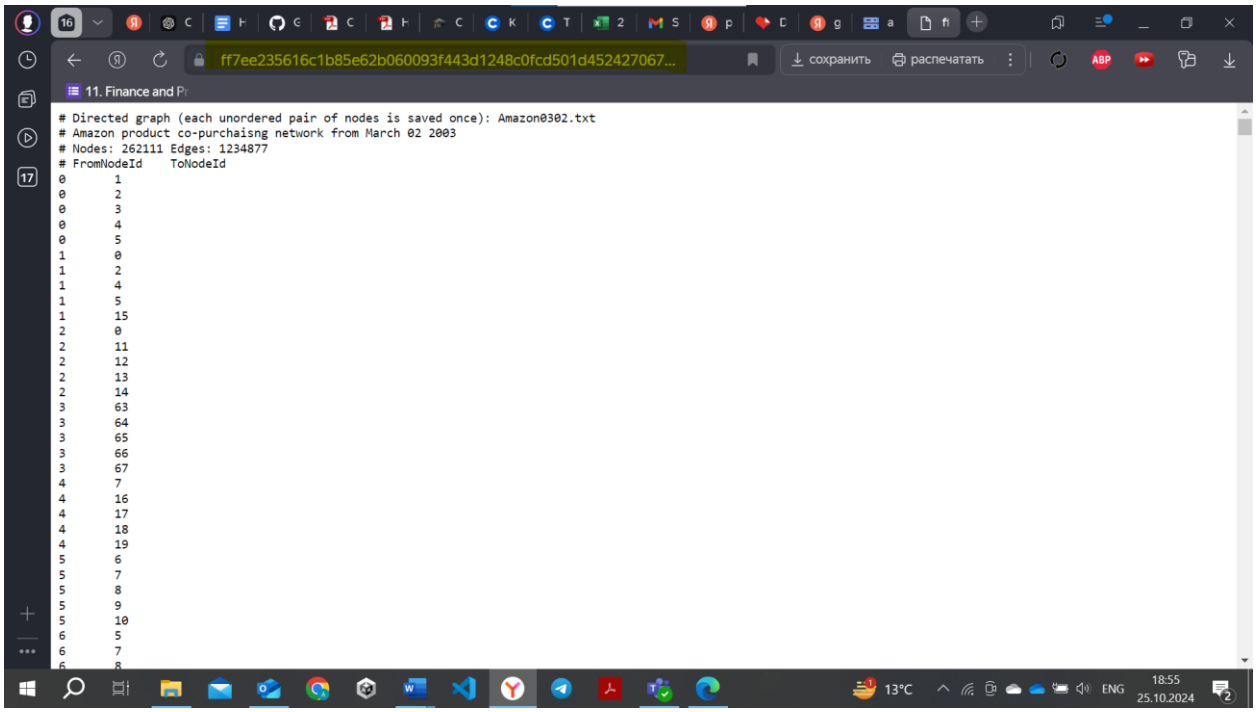
The file management system in Google Cloud Storage enables secure, scalable, and organized file handling for users. Here's how it's set up:

Uploading Files: Users can upload files through the application's backend, which interacts with Google Cloud Storage. Files are assigned unique names or stored in specific folders to avoid conflicts. Each file is uploaded to a designated Cloud Storage bucket, configured with permissions to control access.

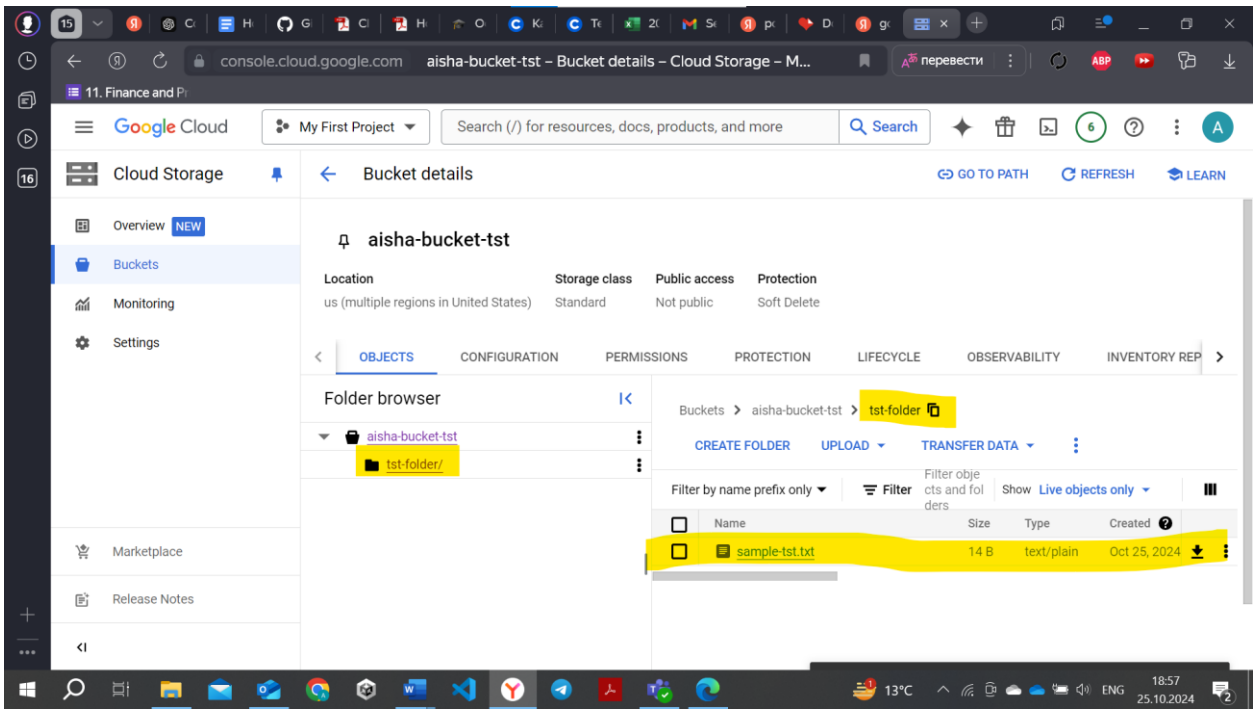


Downloading Files: To download files, the application generates secure, time-limited signed URLs. This ensures that users access files through temporary links, protecting file security and controlling external access.





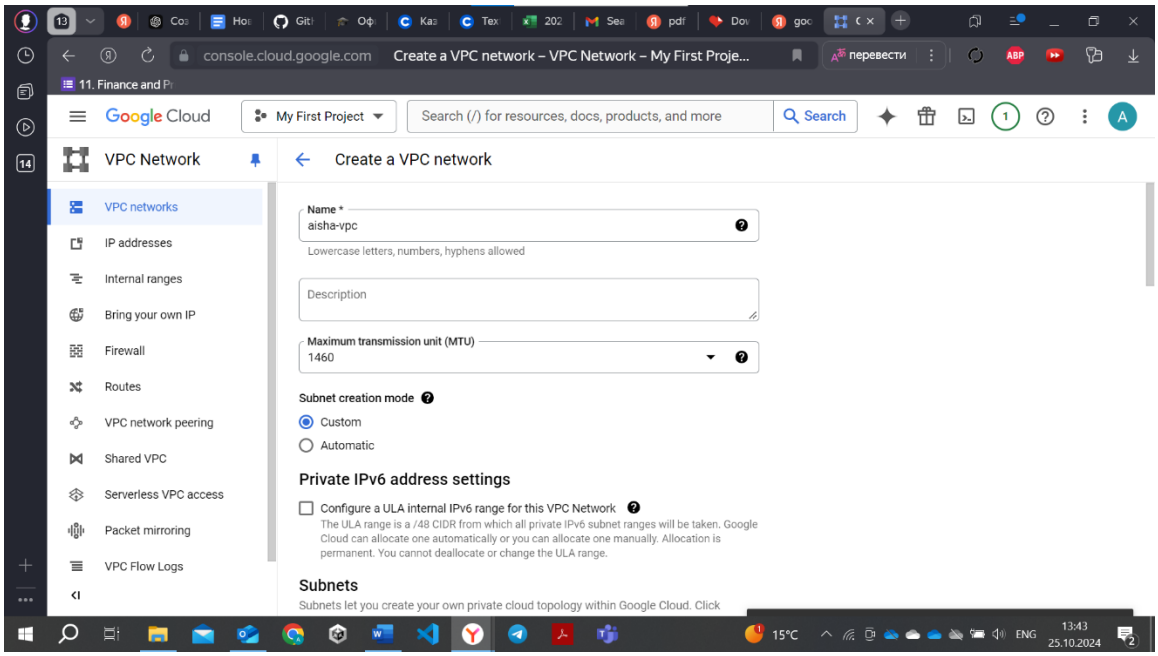
Managing Files: To keep storage organized, files are categorized into folders based on criteria like user ID or project type. Additionally, Object Lifecycle Management policies are enabled to delete or archive older files automatically, managing storage costs and avoiding storage bloat over time. The system also uses Google IAM (Identity and Access Management) roles to assign specific permissions, ensuring each user has the correct access level.

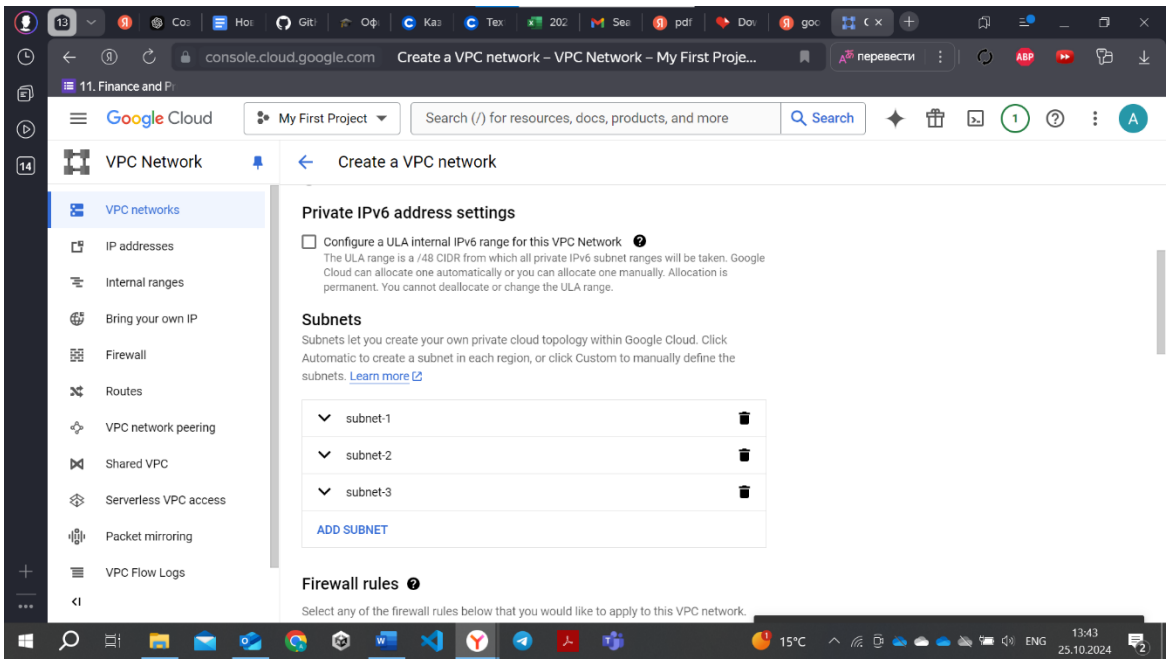


8. Networking in Google Cloud

- ***VPC Creation***: *Detail the creation and configuration of a Virtual Private Cloud (VPC) for the application.*

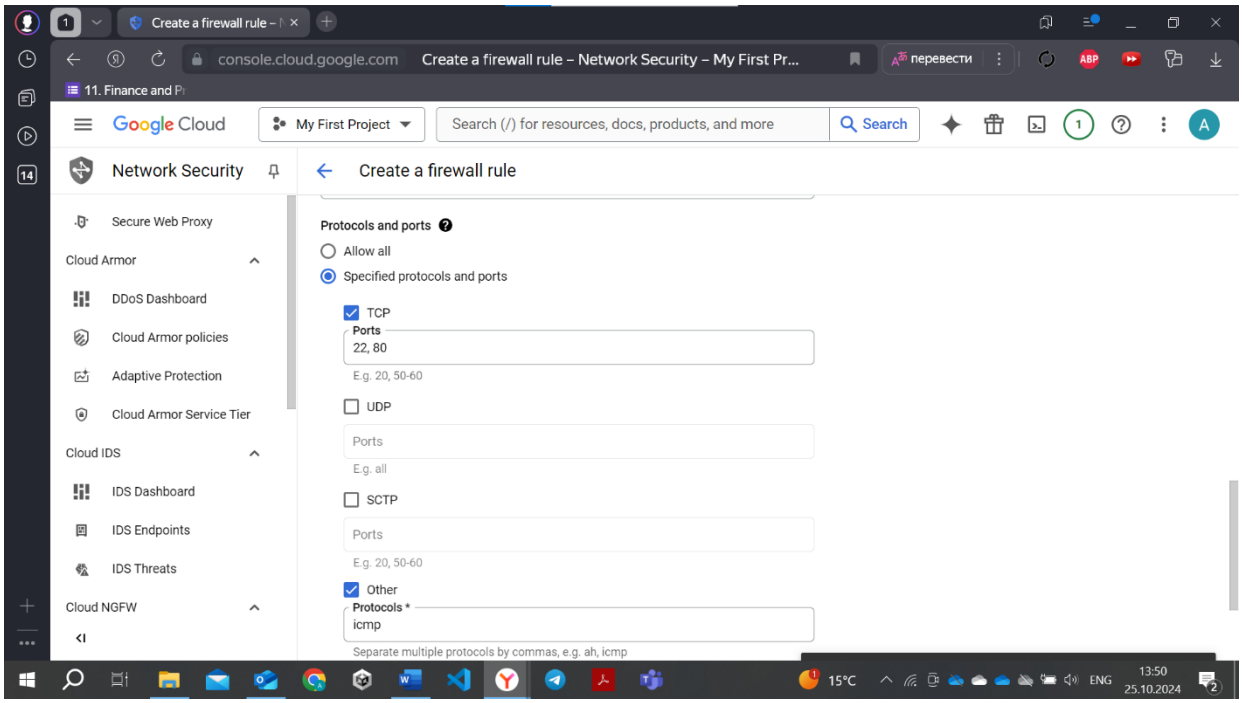
To set up a secure network environment for the application, I created a Virtual Private Cloud (VPC) in Google Cloud. Here's how it was configured:
In the Google Cloud Console, I navigated to the VPC Network section and created a new VPC, naming it specifically for the application `aisha-vpc`. Custom mode was chosen to allow manual control over IP address ranges and subnet configurations.
Three subnets were created within the VPC, each in a different region to support application availability and scalability. Each subnet was assigned a unique IP range to ensure they could communicate without IP conflicts.

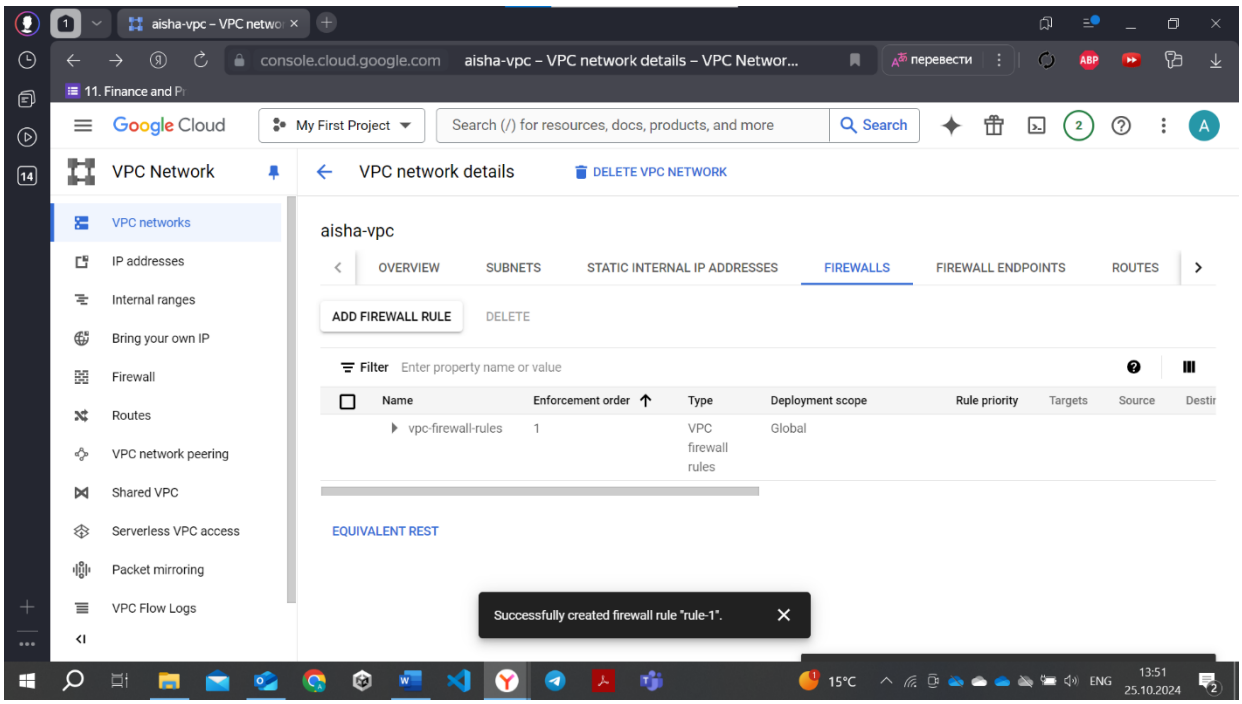




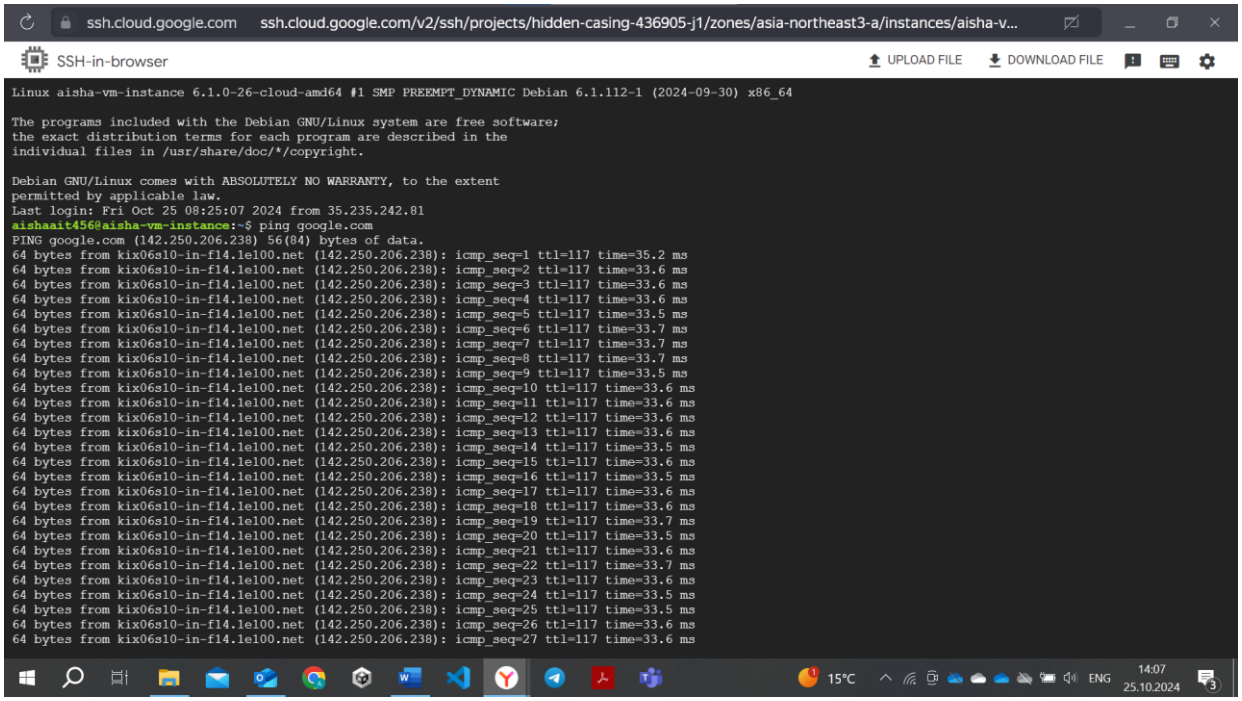
- ***Networking Security****: Discuss the firewall rules and load balancing set up to ensure secure and optimized networking.*

Firewall Rules: Custom firewall rules were added to control incoming and outgoing traffic. For secure access, only specific ports like SSH, HTTP, and HTTPS were opened, limiting traffic to trusted sources. This configuration ensures that only necessary and secure protocols are accessible from the internet.





Internet Access: To allow the application VM to connect to external services, NAT (Network Address Translation) was configured for internet access without exposing internal resources. This allowed for outbound internet connectivity for updates and API calls, while keeping VMs securely shielded.

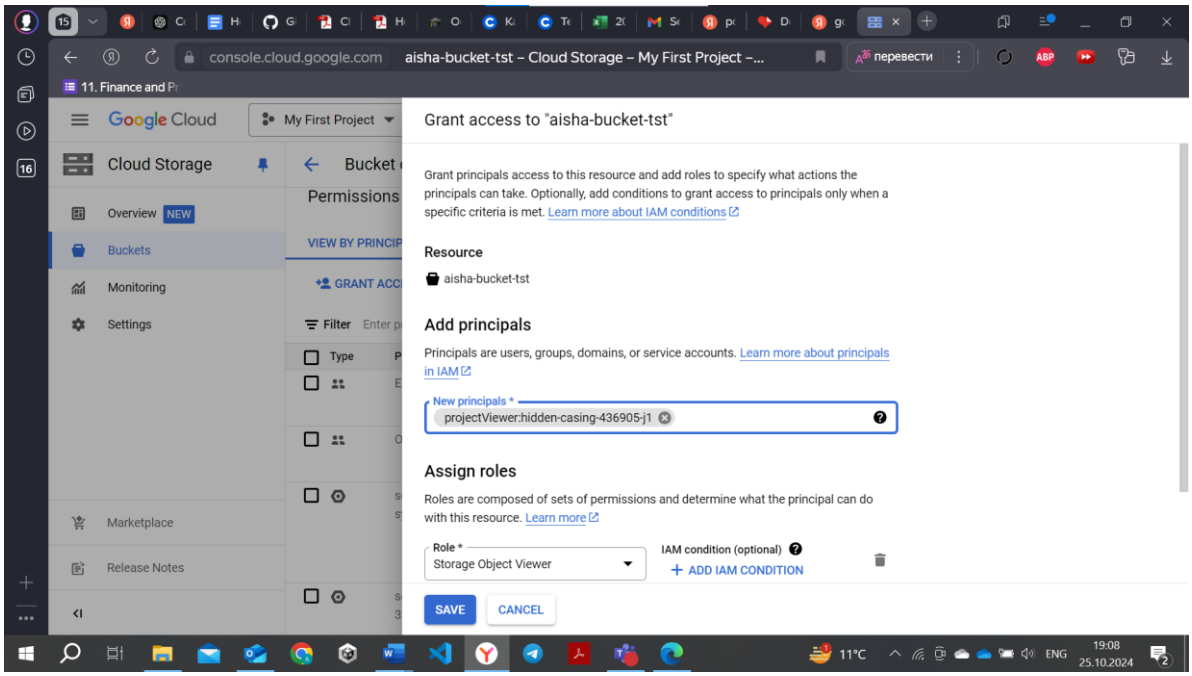


This VPC setup provides a secure, isolated environment with controlled network access, making it ideal for a cloud-based application.

9. Identity and Security Management

- ***IAM Implementation**: Explain how Identity and Access Management (IAM) was set up to manage user roles and permissions.*

To manage user access securely, Identity and Access Management (IAM) roles were configured in Google Cloud to control permissions for each part of the application. Specific roles were assigned based on user requirements, such as "Viewer" for read-only access and "Editor" for those needing full access to manage resources. Custom roles were also created to restrict access to sensitive actions, ensuring that each user or service has only the permissions necessary to perform their tasks.



- ***Security Measures****: Describe the measures taken to ensure data security, such as encryption and access controls.*

Multiple security practices were implemented to protect data within the application. Data is encrypted both at rest and in transit using Google Cloud's built-in encryption, ensuring that files remain secure and private. IAM roles and permissions were carefully assigned to prevent unauthorized access, with logging enabled for all access attempts. Additionally, firewall rules restrict network traffic, and Google Cloud's security policies are in place to prevent data breaches. These combined measures provide a strong security framework for managing sensitive data in the cloud.

10. Testing and Quality Assurance

Outline the testing methodologies used, including unit tests, integration tests, and any load testing performed to evaluate performance.

To ensure the application is reliable, secure, and performs well, several testing methodologies were applied:

- **Unit Testing**: Individual components, like the backend API and storage integration, were tested using unit tests. This ensured each function performed as expected and handled both standard and edge cases accurately.
- **Integration Testing**: Integration tests were conducted to verify seamless interactions between services like Compute Engine, Cloud Storage, and the VPC network. These tests

focused on ensuring that data was properly uploaded, stored, and retrieved without errors, confirming end-to-end functionality.

- **Load Testing**: Load testing simulated high user activity to evaluate the system's performance and scalability. This was crucial to identify any potential bottlenecks, particularly in the VM's handling of concurrent requests and storage response times. Results from these tests were used to fine-tune resource allocation and firewall settings to handle peak loads efficiently.

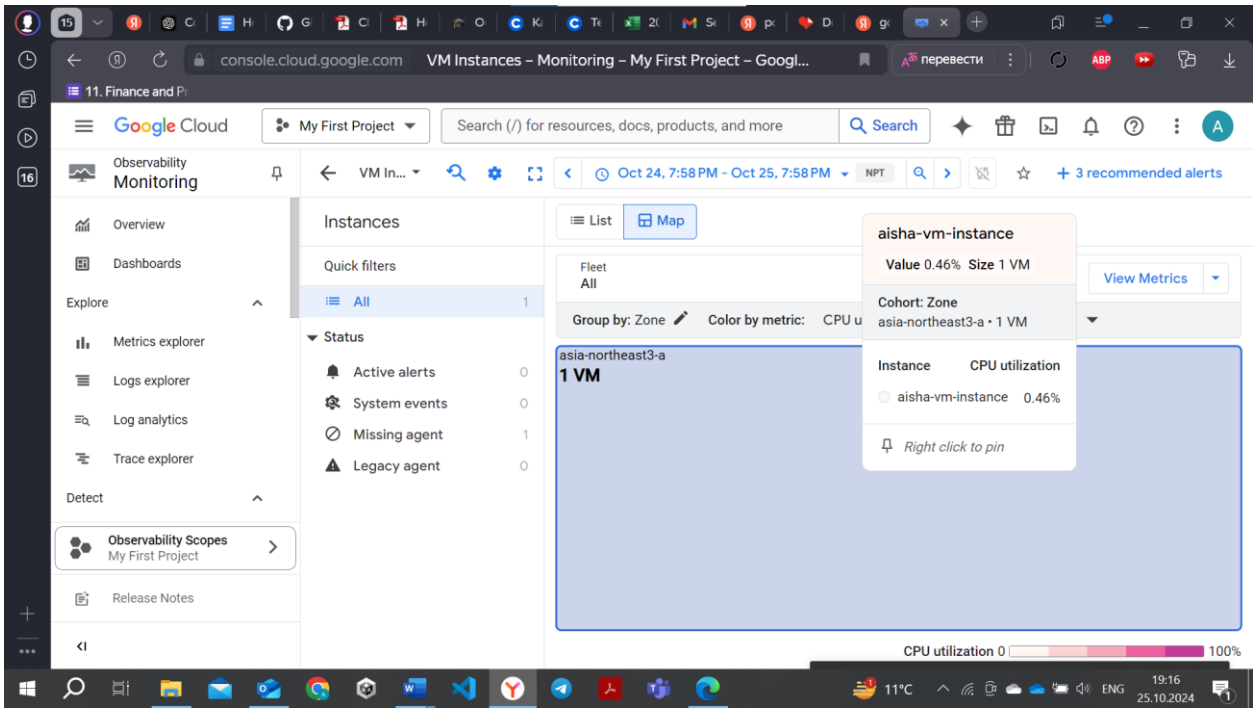
These testing methods helped validate the application's robustness, making sure it could handle real-world usage and scaling requirements effectively.

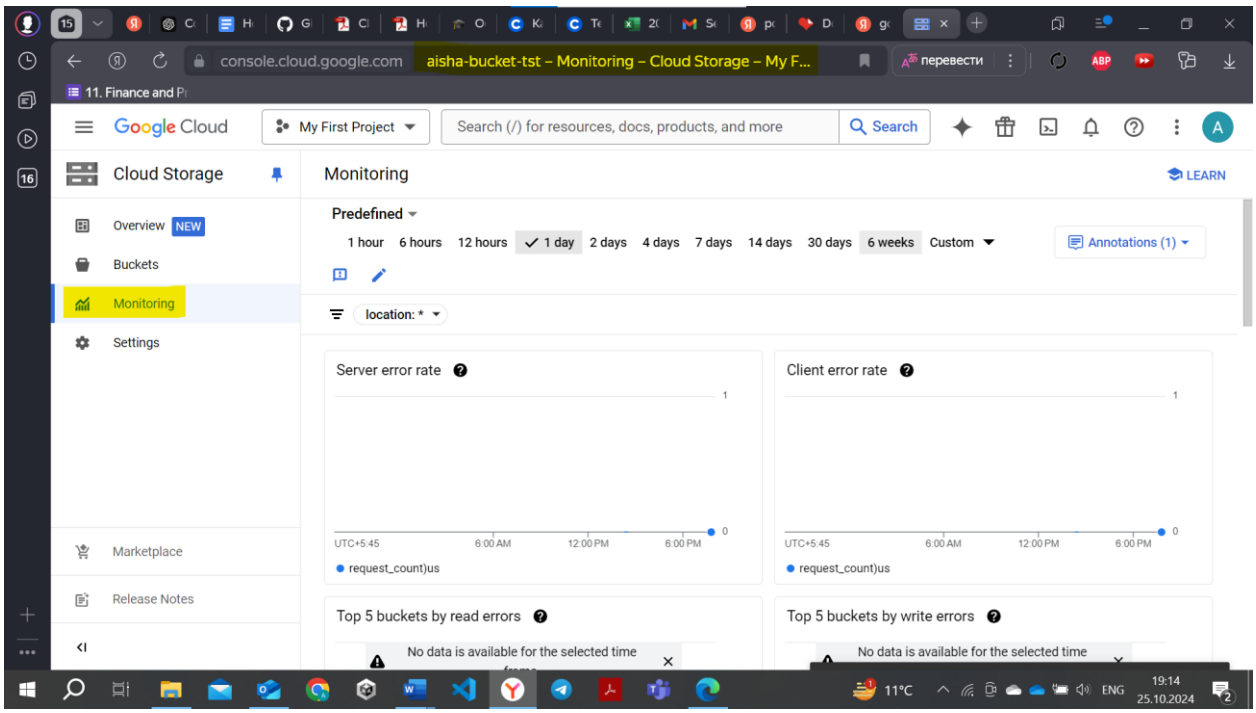
11. Monitoring and Maintenance

Discuss the monitoring tools used to track application performance and health, and the strategies established for maintenance.

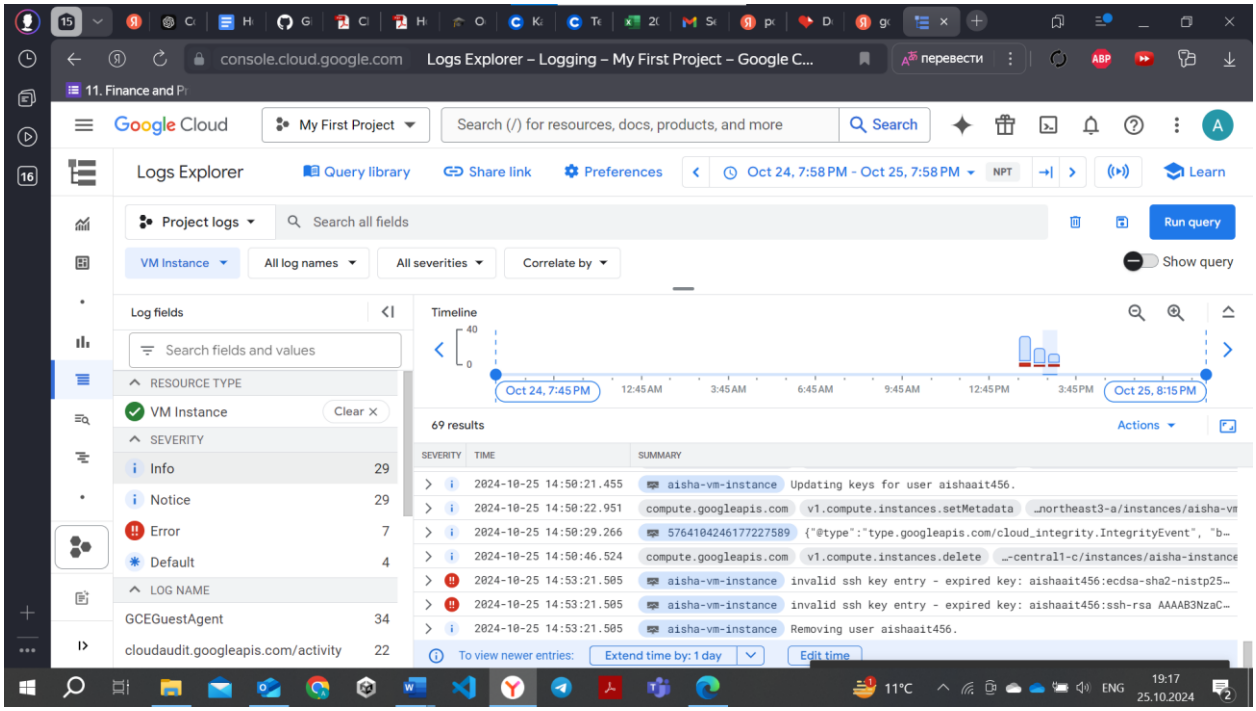
To ensure the application's ongoing performance and health, I implemented monitoring and maintenance strategies using Google Cloud's tools:

- **Monitoring Tools**: Google Cloud Monitoring was configured to track key metrics like CPU usage, memory, disk space, and network traffic for virtual machines and storage activity. Alerts were set up to notify when resource usage exceeded certain thresholds, enabling proactive response to potential issues before they impacted users.

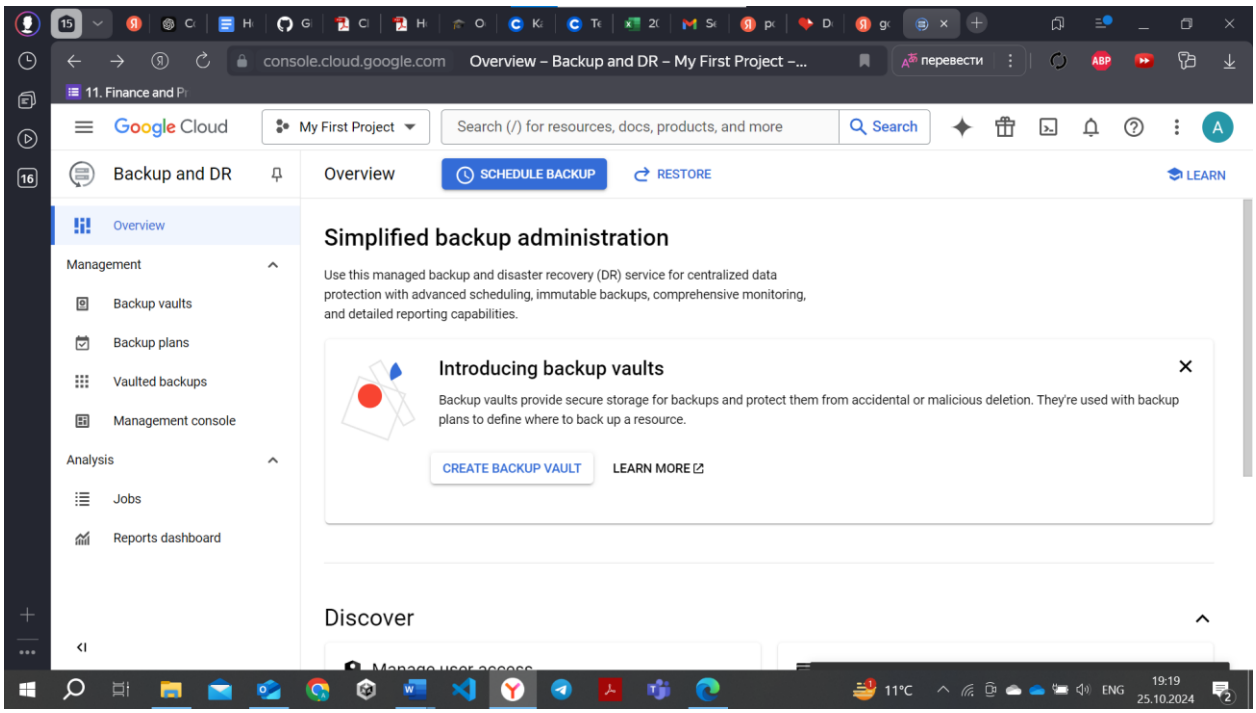




- **Logging and Analysis**: Google Cloud Logging was used to collect logs from the application, tracking events like user activity, API requests, and system errors. This logging allowed for quick troubleshooting and performance tuning based on real-time usage patterns.



- **Maintenance Strategy**: Routine maintenance can include regular backups of essential data, updates to firewall rules, and IAM role reviews to ensure security remained tight. Scheduled VM reboots and storage checks can also be performed to ensure stability and avoid downtime.



These combined strategies help keep the application reliable and responsive, supporting long-term performance and security.

12. Challenges and Solutions

Summarize the significant challenges faced during the project and the solutions implemented to overcome them.

During this project, several challenges emerged, each requiring specific solutions:

- **Challenge: Network Configuration Complexity** – Setting up a secure and accessible network was challenging, particularly configuring the VPC and firewall rules to allow necessary traffic while restricting unauthorized access.
    - **Solution**: I carefully planned and tested firewall rules, applying the principle of least privilege to limit open ports and traffic sources. Additional logging was enabled to monitor access and adjust configurations as needed.
- **Challenge: Managing User Access with IAM** – Assigning precise access rights without over-permissioning was difficult, especially with multiple user roles involved.
    - **Solution**: To streamline access control, custom IAM roles were created, with permissions tailored to each user's tasks. This approach minimized risk while supporting necessary access.
- **Challenge: Application Scalability Under Load Testing** – Initial load tests showed performance slowdowns when handling high volumes of requests.
    - **Solution**: Resource allocation was optimized by resizing the VM and enabling autoscaling, allowing the system to handle high traffic more efficiently.

Each of these challenges was addressed through careful planning, testing, and iterative improvements, ultimately creating a more resilient and scalable cloud-based application.

13. Conclusion

Reflect on the project outcomes, the effectiveness of the technologies used, and suggestions for future enhancements.

This project successfully demonstrated the power and flexibility of Google Cloud Platform for building a secure and scalable file-sharing application. The combination of GCP services—including Compute Engine, Cloud Storage, and VPC—allowed for robust storage, reliable networking, and efficient user management, proving effective for our needs. The security measures and IAM roles ensured data safety, while monitoring tools provided valuable insights for maintenance and performance tracking.

While the current setup is stable and responsive, future enhancements could focus on automating more tasks, such as automated scaling configurations and incorporating additional security layers like DLP (Data Loss Prevention). Overall, this project highlights how GCP's integrated services support efficient, flexible cloud solutions and provides a strong foundation for future growth.

14. References

Google Cloud Documentation:

- Compute Engine: Google Cloud Compute Engine Documentation
- Cloud Storage: Google Cloud Storage Documentation
- VPC Networking: Google Cloud VPC Documentation
- IAM: Google Cloud IAM Documentation
- Monitoring and Logging: Google Cloud Monitoring and Logging

Cloud Computing Concepts:

- IBM Cloud Learn Hub: What is Cloud Computing?
- Microsoft Azure Documentation: Introduction to Cloud Computing

Online Tutorials and Resources:

- Coursera: Google Cloud Platform Fundamentals
- YouTube: "Introduction to Google Cloud Platform" by Google Cloud Platform Channel
- GCP Free Tier and Pricing Guide: Google Cloud Pricing

15. Appendices

Include any additional diagrams, code snippets, or data that support the report but are too detailed for the main sections.

All necessary screenshots, code snippets, and other relevant materials are included above in the report. Therefore, I believe no additional appendices are required.