FINAL PROJECT

CLOUD COMPUTING (BASIC)

Giulio Fantuzzi

PROBLEM OVERVIEW

PROJECT AIMS

- To identify and deploy a cloud-based file storage system
- To enable users to upload, download and delete files
- To address scalability, security and cost-efficiency
- To test the platform under stress

WHY NEXTCLOUD?

- Simplicity in implementing the system
- Supported by an extensive documentation
- Being open-source, it benefits from a vast community of developers and users
- Comprehensive set of features beyond basic file storage

```
📁 CloudBasic_FinalProject/
│
├──── 📄 create_usr.sh
│
├──── 📁 data/
│    └──── ...files that will be created...
│
├──── 📄 delete_usr.sh
│
├──── 🐳 docker-compose.yml
│
├──── 📄 generate_files.sh
│
├──── 📁 locust-tests/
│    └──── 🐍 locustfile.py
│
├──── 📰 README.md
│
└──── 🏗 setup.sh
```

USER AUTHENTICATION AND FILE OPERATIONS

Requirements about user authentication and authorization are guaranteed:

User Management

- Nextcloud offers a built-in user management system
  - sign-up
  - login
  - logout
- User-friendly system, ensuring users to easily navigate the platform

Role Based Access Control (RBAC)

- Different user roles:
  - Regular users
  - Admins
- Crucial to manage access levels in the system
- Admins have permissions to manage users
- Regular users have their private storage space

Admin Management

- Admins manage users through an admin interface
  - Create users
  - Edit users
  - Delete users
  - Assigning/Change roles and permissions

Private Storage Space

- Users are automatically assigned a private storage space upon account creation
- Default 512 MB storage quota-per-user
- Admins can adjust individual users' quotas through the web interface
- Global quota by modifying a *config.php* file

ADDRESS SECURITY

- Nextcloud offers a comprehensive suite of security features that can be configured from the administrators interface
- All the options can be found in **Administration settings/Security/** and can be easily enabled
- Despite not being essential for the purposes of this exercises, they would be crucial for a real-world scenario

Basic password policies

- Minimum password length
- User password history
- Number of days until a user password expire
- N. of login attempts before an account is blocked

Enhanced password policies

- Forbid common passwords
- Enforce upper and lower case characters
- Enforce numeric/special characters
- Check passwords against the list of breached passwords from *haveibeenpwned.com*

Server-side-encryption (SSE)

- Files are encrypted before being stored
- Files automatically encrypted upon upload
- Files automatically decrypted upon download
- Encryption process transparent to the end-users
- Some drawbacks in terms of performance, but fundamental to strengthen the overall security

Two factor authentication (2FA)

- Adds an additional layer of protection
- It enforces users to provide two forms of identification: a password + a second factor
- It reduces unauthorized access to accounts
- Deliberately excluded from my analysis....

COST-EFFICIENCY

- I referred to Nextcloud Server, which is the **free version** of Nextcloud
- It can be deployed on a dedicated server infrastructure, making it particularly suitable for small/medium businesses
- **Nextcloud Enterprise** for Larger organizations with more advanced needs and requiring professional support

2 Sources of costs

Hardware on which Nextcloud is hosted

Storage solutions used to store the data

- **Most intriguing challenge:** cost-efficiency in **storage management**
  - Organizations experiencing fluctuating data usage and uncertain business trajectory (startups)
  - Forecasting the necessary amount of storage often proves to be infeasible
- **Balanced approach** that incorporates both on-premises and cloud storage solutions:
  - On-premises storage to perform essential tasks independently of external providers
  - Cloud storage solution through a *pay-as-you-go* model according to demand

ADDRESS SCALABILITY

- As the number of users and files increases, it becomes essential to implement a scalable file storage architecture
- Solution 1: **mixed strategy** combining on-premises and cloud storage solutions
- Solution 2: deploying a distributed environment that employs **horizontal partitioning of data**
    - Viable approach to enhance scalability
    - Data divided into smaller, more manageable chunks
    - Load-data distributed across multiple nodes and improving performance

- What about **NoSQL solutions** ?
    - In a distributed setup, NoSQL solutions like MongoDB may be intriguing
    - Scalability, flexibility, and performance for handling large volumes of unstructured data
- BUT feasibility not trivial at all
    - Nextcloud does not directly support NoSQL databases🥲(it primarily works with SQL databases)
    - Transition to a NoSQL database is strictly dependent on some data requirements ⚠

- I equipped my Nextcloud instance with **Redis** a caching mechanism which allows to store frequently accessed data into memory, reducing the need to query the primary storage or database for every request

mongo DB

n8n.io

DEPLOYMENT OF THE SYSTEM

The deployment of Nextcloud has been executed using **Docker** and **Docker Compose**

Docker images:

- **nextcloud**
  - To deploy the Nextcloud application
  - Configured with environment variables for database connection details, admin user credentials, and other settings
- **mariadb**
  - To deploy a MariaDB database server, providing the backend storage for Nextcloud
  - Configured with environment variables for setting up the root password, db name and user credentials
- **redis**
  - This image sets up a Redis server, which is utilized for caching purposes within the Nextcloud application
- **locustio/locust**
  - Open-source load testing tool, which is used to simulate user traffic and analyze the performance
  - Configured to run tests defined in a *locustfile.py* script targeting the Nextcloud instance.

EXTERNAL CLOUD PROVIDER TO DEPLOY THE SYSTEM

For deploying a Nextcloud instance in a production environment, I would opt for **AWS (Amazon Web Services)** since it emerges as a compelling choice due to its scalability, reliability, and the wide range of services that it offers

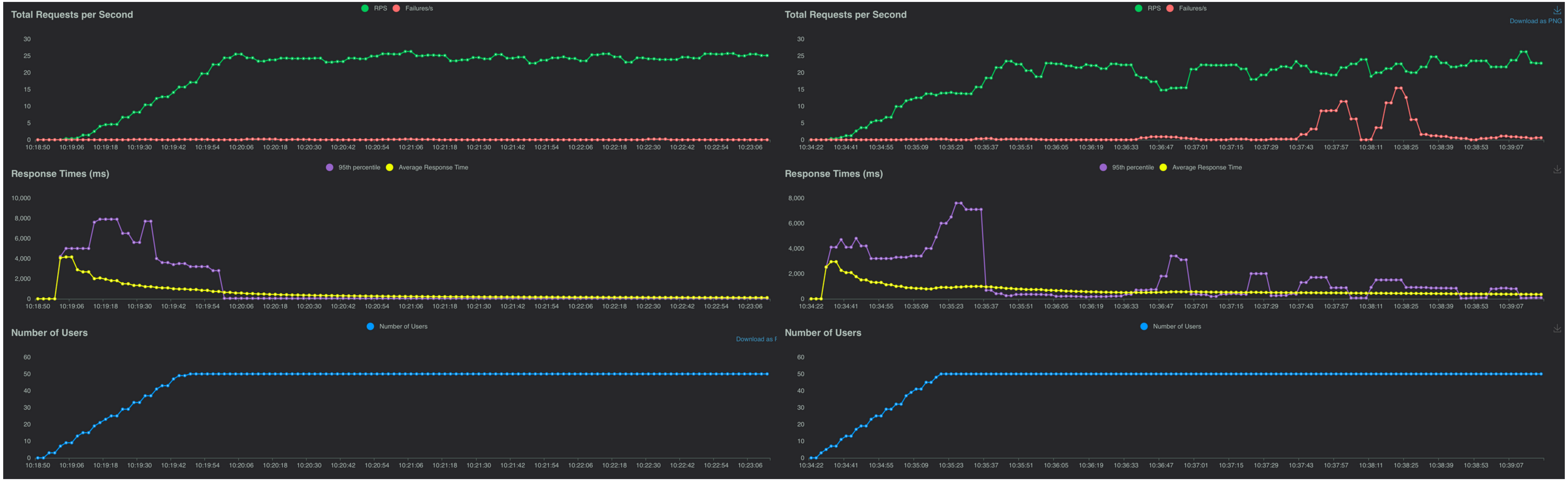
- **Scalability and Performance**
  - Highly scalable infrastructure, allowing to accommodate the growth as increase:
    - the user base
    - the storage needs
  - Services as Amazon S3, which is highly scalable and can serve thousands of HTTP requests per second
  - Crucial for handling varying loads and ensuring the Nextcloud instance remains responsive and available to users
- **Reliability**
  - High reliability and availability
  - Data centers located in various regions worldwide ⟶ Nextcloud instance remains accessible and functional
- **Cost-Effectiveness**
  - *pay-as-you-go* model, which can lead to significant cost savings (already discussed)
- **Integration**
  - AWS offers integration with a vast ecosystem of services and tools

LOCUST TESTING

All locust tasks were defined in a Python script and executed on my MacBook Pro M2 laptop

Tests to assess the system's performance under varying stress conditions

Pool of 50 users


Total Requests per Second
● RPS ● Failures/s
10:18:50 10:19:06 10:19:18 10:19:30 10:19:42 10:19:54 10:20:06 10:20:18 10:20:30 10:20:42 10:20:54 10:21:06 10:21:18 10:21:30 10:21:42 10:21:54 10:22:06 10:22:18 10:22:30 10:22:42 10:22:54 10:23:06
Response Times (ms)
● 95th percentile ● Average Response Time
Number of Users
● Number of Users
Download as F
Total Requests per Second
● RPS ● Failures/s
Download as PNG
10:34:22 10:34:41 10:34:55 10:35:09 10:35:23 10:35:37 10:35:51 10:36:05 10:36:19 10:36:33 10:36:47 10:37:01 10:37:15 10:37:29 10:37:43 10:37:57 10:38:11 10:38:25 10:38:39 10:38:53 10:39:07
Response Times (ms)
● 95th percentile ● Average Response Time
Number of Users
● Number of Users