Decentralized File Exchange Hub: A Cloud-Native Approach

Rafael Pereira[a]

[a]Computer Science and Communications Research Centre, School of Technology and Management, Polytechnic of Leiria, 2411-901 Leiria, Portugal

ARTICLE INFO

ABSTRACT

In this report, we present a decentralized file exchange hub designed using a cloud-native approach that leverages containers, microservices, and cloud services. Our architecture comprises a React-based frontend served by Nginx, an Express API for managing file transfers, a real-time communication server using Socket.IO, and MongoDB as the primary database. The application stores files in Google Cloud Storage, and our deployment pipeline is streamlined using Google Cloud Build and Cloud Run. The result is a scalable, resilient, and efficient file exchange system that demonstrates the power and flexibility of modern cloud computing technologies for software applications.

1. Introduction

In recent years, cloud computing has revolutionized the way we develop, deploy, and manage software applications. This report presents a decentralized file exchange hub built with a cloud-native approach, utilizing containers, microservices, and cloud services. Our project focuses on leveraging the power of modern cloud technologies to create a scalable, resilient, and efficient file exchange system.

2. Architecture

This project follows a microservices architecture, which allows for improved scalability, resilience, and maintainability. The system is divided into four main components, each responsible for a specific aspect of the application. Figure 1 presents a diagram of the architecture, illustrating the relationship between each component.

The following UML diagram (Figure 1) depicts the relationships between the components:

In this architecture, each component communicates with the others through well-defined interfaces, allowing for efficient collaboration and easy maintenance. The use of containerization and cloud-native deployment further enhances the system's scalability and resilience.

2.1. Presentation Provider: Nginx-based Frontend

The presentation layer of our file exchange hub is built using a React web application. We utilize Nginx as a reverse proxy and web server to serve the static content generated by the React application. Nginx offers excellent performance, stability, and low resource usage, making it an ideal choice for our frontend.

2.2. API: Express-based File Transfer Service

Our file transfer service is implemented using Node.js with the Express framework, providing a robust and scalable API for managing file uploads and downloads. This service is responsible for handling file metadata, coordinating file
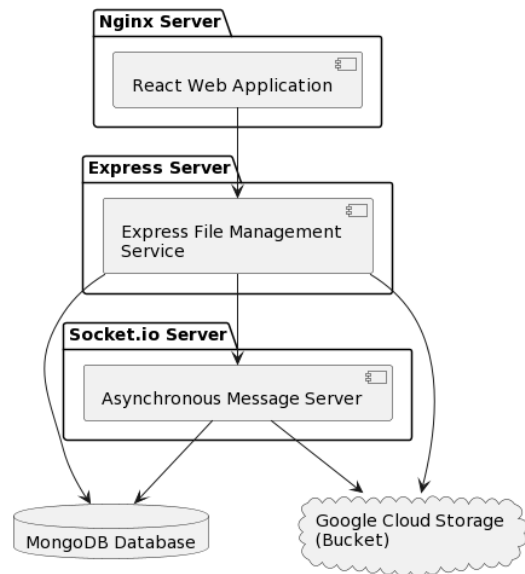
✉ rafael.m.pereira@ipleiria.pt (R. Pereira)
ORCID(s): 0000-0001-8313-7253 (R. Pereira)
🔗 https://www.linkedin.com/profile/view?id='rafaelmendespereira' (R. Pereira)

Figure 1: System architecture diagram

transfers, and integrating with the cloud storage service for actual file storage.

2.3. Asynchronous Message Communication Server: Socket.IO

To enable real-time communication and notifications within our file exchange hub, we employ Socket.IO, a popular library for real-time web applications. This server facilitates instant messaging between users in the same room, ensuring a seamless and interactive experience for all participants.

2.4. Database: MongoDB

MongoDB, a NoSQL database, is utilized to store information about rooms, files, and users. Its flexible schema and horizontal scalability make it a fitting choice for our cloud-native application, as it can easily adapt to changing requirements and handle large volumes of data.

2.5. Cloud Storage Bucket: Google Cloud Storage

To store and manage the actual files exchanged by users, we leverage Google Cloud Storage. This fully managed object storage service offers high availability, durability, and scalability, allowing our file exchange hub to efficiently store, retrieve, and share files.

3. Deployment

The deployment process is designed to ensure a smooth transition from development to production, leveraging the power of Google Cloud Platform (GCP) services. The deployment strategy uses GCP's Cloud Build and Cloud Run services for automating and streamlining the process. Figure 2 presents a diagram of the deployment process, illustrating the steps and relationships between each involved service.

The following UML diagram (Figure 2) depicts the deployment process and the relationships between the components:
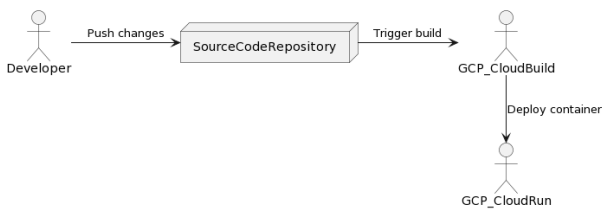

Figure 2: Deployment process diagram

The deployment process ensures that the application components are consistently built and deployed across environments, minimizing human error and facilitating continuous integration and delivery.

3.1. Cloud Build: Continuous Integration and Deployment

For a streamlined development and deployment process, we use Google Cloud Build, a managed service for building, testing, and deploying container images. This service automates the process of building Docker images from our source code and deploying them to Google Cloud Run, ensuring a consistent and reliable deployment pipeline.

3.2. Cloud Run: Scalable Containerized Services

Our application's microservices are containerized and deployed on Google Cloud Run, a serverless platform that runs Docker containers. This allows us to easily scale our services up or down based on demand, ensuring optimal resource usage and cost-efficiency.

4. Conclusion

This report has presented a decentralized file exchange hub designed with a cloud-native approach. By leveraging containers, microservices, and cloud services, we have created a scalable, resilient, and efficient system for exchanging files. Our use of modern technologies such as Nginx, Express, Socket.IO, MongoDB, and Google Cloud services demonstrates the power and flexibility of cloud computing for modern software applications.

CRediT authorship contribution statement

: Conceptualization of this study, Methodology, Software.

References

My name is Rafael Pereira, I'm 22 years old, and I'm an MSc Computer Engineering Student and Researcher at Polytechnic of Leiria. I'm extremely curious about the technology world, ambitious for knowledge in this area, I'm dedicated to doing my work. The programming thing always got me, and every day it grows. Today I'm looking for interesting projects, that can make me think and learn every day.