DR. BABASAHEB AMBEDKAR TECHNOLOGICAL UNIVERSITY, LONERE.

PROJECT REPOET PHASE 1

Project report on

"GCP FILESTORE"

M.B.E.SOCIETY'S
COLLEGE OF ENGINEERING,AMBAJOGAI.

Department Of Final Year Computer Science And Engineering

2024-2025

Under the Nobel Guidance of

Prof. Mr. V. M. CHANDODE

Submitted By

11. GANESH KALE
26. ATHARVA SALUNKE
39. DNYANESHWAR GITTE
54. VAISHNAVI INGALE

DR. BABASAHEB AMBEDKAR TECHNOLOGICAL UNIVERSITY, LONERE.

M.B.E.SOCIETY'S COLLEGE OF ENGINEERING,AMBAJOGAI.

Project report on

"GCP FILESTORE"

Under the Nobel Guidance of
Prof. Mr. V. M. CHANDODE

Department Of Final Year Computer Science And Engineering
2024-2025

Submitted By

11. GANESH KALE
26. ATHARVA SALUNKE
39. DNYANESHWAR GITTE
54. VAISHNAVI INGALE

DR. BABASAHEB AMBEDKAR TECHNOLOGICAL UNIVERSITY, LONERE.

M.B.E.SOCIETY'S COLLEGE OF ENGINEERING,AMBAJOGAI.

Certificate

This is to certify that "Ganesh Kale,Atharva Salunke,Dnyaneshwar Gitte,Vaishnsvi Ingale" have completed project phase- I on "**GCP FILESTORE** " in partial fulfillment of the requirement for the degree in Bachelor of Technology (B. Tech.) in Computer Science and Engineering of Dr. Babasaheb Ambedkar Technological University, Lonere, during the academic year "2024 2025".

Prof.V.M. Chandode
Guide

Prof.S.V.Kulkarni
Head of Dept.

Dr. Nandkumar Rawbawale
Principal

"ACKNOWLEDGEMENT"

We Offer my sincere and hearty thanks, with a deep sense of gratitude to my guide **Prof. V.M. Chandode** and head of department **Prof. S.V. Kulkarni** for their valuable direction and guidance to my project *"**GCP FILESTORE**"* and their meticulous attention to my seminar work without taking care of their voluminous work.

We thankful to our principal **Dr. Nandkumar Rawbawale** for his encouragement towards my project.

Last but not least I am thankful to my friends and well wishers, to whom I am indebted for their constant help, encouragement and without whom this project would not have been a success.

Mr.GANESH KALE.
Mr.ATHARVA SALUNKE
Mr.DNYANESHWAR GITTE.
Ms.VAISHNAVI INGALE.

CONTENT

Sr. No.	TOPIC	PAGE No.
1)	**ABSTRACTION**	6
2)	**INTRODUCTION (GCP)**	7
3)	**PROBLEM STATEMENT**	8
4)	**PROBLEM TO SOLUTION**	9
5)	**IMPLEMENTATION**	10
6)	**FEATURES AND FUNCTIONALITIES**	12
7)	**SYSTEM DESIGN AND ARCHITECTURE**	14
8)	**FUTURE ENHANCEMENT**	16
9)	**CONCLUSION**	17
10)	**REFERENCES**	18

1.ABSTRACTION

Cloud computing has revolutionized how businesses and individuals store, manage, and access data. A critical part of this ecosystem is cloud-based file storage, which ensures secure, scalable, and reliable storage solutions[5]. The **GCP File Storage System Project** leverages **Google Cloud Filestore**, a managed network-attached storage (NAS) service, to provide a robust file storage platform accessible via web-based technologies[2].

This project aims to develop a user-friendly system that allows individuals and organizations to upload, manage, and retrieve files efficiently. By combining web technologies like HTML, CSS, JavaScript, and backend frameworks with Google Cloud services, the system offers high-performance storage with seamless accessibility[3]. The project focuses on delivering scalable file storage to meet diverse demands, from individual users to enterprise-level applications, while ensuring data security and compliance.

The architecture is built around GCP's core services. **Google Cloud Filestore** provides the primary storage, offering high IOPS and low latency, crucial for handling large files and frequent access. **Google Kubernetes Engine (GKE)** or **Cloud Functions** powers the backend logic, while **Firebase Authentication** ensures secure user access[1]. Data protection mechanisms, such as encryption and backups, guarantee data integrity and recovery in case of failures.

From a user perspective, the system provides an intuitive web interface where users can easily upload, view, and manage files. Advanced functionalities like role-based access control, real-time file synchronization, and comprehensive activity logs are implemented for enhanced usability[6]. Developers and businesses can also integrate the system into their workflows using RESTful APIs, making it versatile for varied use cases.

In addition to its core functionality, this project incorporates scalability and disaster recovery, ensuring uninterrupted service even under heavy workloads or regional failures. This reliability is achieved using GCP's global infrastructure, which supports rapid scaling and robust failover mechanisms.

The GCP File Storage System project is not only a technical endeavor but also a demonstration of modern cloud capabilities [7]. It highlights the power of serverless technologies, containerization, and managed services in building efficient systems. Furthermore, it emphasizes sustainability, as cloud-based solutions reduce on-premises infrastructure costs and energy consumption.

In conclusion, the GCP File Storage System bridges the gap between technical complexity and user convenience. By leveraging state-of-the-art technologies, the project delivers a solution tailored to meet contemporary storage demands while setting a foundation for future enhancements.

2.INTRODUCTION (GCP)

a. Google Cloud Platform (GCP) has emerged as a cornerstone of modern cloud computing, offering a robust ecosystem for building, deploying, and managing scalable applications. Among its suite of services, **Google Cloud Filestore** provides managed file storage, ideal for handling high-performance, shared storage needs. This project demonstrates the integration of GCP services to create a reliable file storage solution accessible via a web interface.

b. The **GCP File Storage System Project** leverages Filestore for its scalable and low-latency storage capabilities. Designed for developers, businesses, and end-users, this system enables seamless file uploads, retrievals, and management through a secure and user-friendly platform. Backed by GCP's infrastructure, it supports high IOPS for demanding workloads, making it suitable for a range of applications such as collaboration platforms, web hosting, and content management systems.

c. Central to the project is GCP's ability to scale and provide enterprise-grade reliability. Using services like **Google Kubernetes Engine (GKE)** or **Cloud Functions**, the backend efficiently handles storage operations and user requests. The system also incorporates **Firebase Authentication** for secure user access and **Cloud Storage APIs** for integration with other systems. These technologies ensure that the solution is flexible, extensible, and easy to maintain.

d. What sets GCP apart in this context is its focus on data security and disaster recovery. By offering built-in encryption and zonal redundancy, GCP protects sensitive information and ensures business continuity. The project also uses RESTful APIs to facilitate integrations with third-party applications, enhancing its utility across diverse industries.

e. In summary, GCP demonstrates how advanced cloud services can transform storage systems into scalable, efficient, and secure platforms. By leveraging its ecosystem, the GCP File Storage System project illustrates the potential of cloud-native solutions to meet modern data storage challenges, empowering users with unparalleled accessibility and performance.

3.PROBLEM STATEMENT

Modern organizations and individuals face challenges in securely storing and managing large volumes of data while ensuring accessibility, scalability, and cost-efficiency. Traditional storage systems often struggle with performance bottlenecks, limited scalability, and inadequate disaster recovery solutions. The need for a robust, cloud-based storage system that integrates seamlessly with web technologies has become critical. This project addresses these challenges by leveraging **Google Cloud Filestore** to create a scalable, high-performance, and secure file storage solution, accessible via a user-friendly web interface and supporting seamless integration for diverse applications.

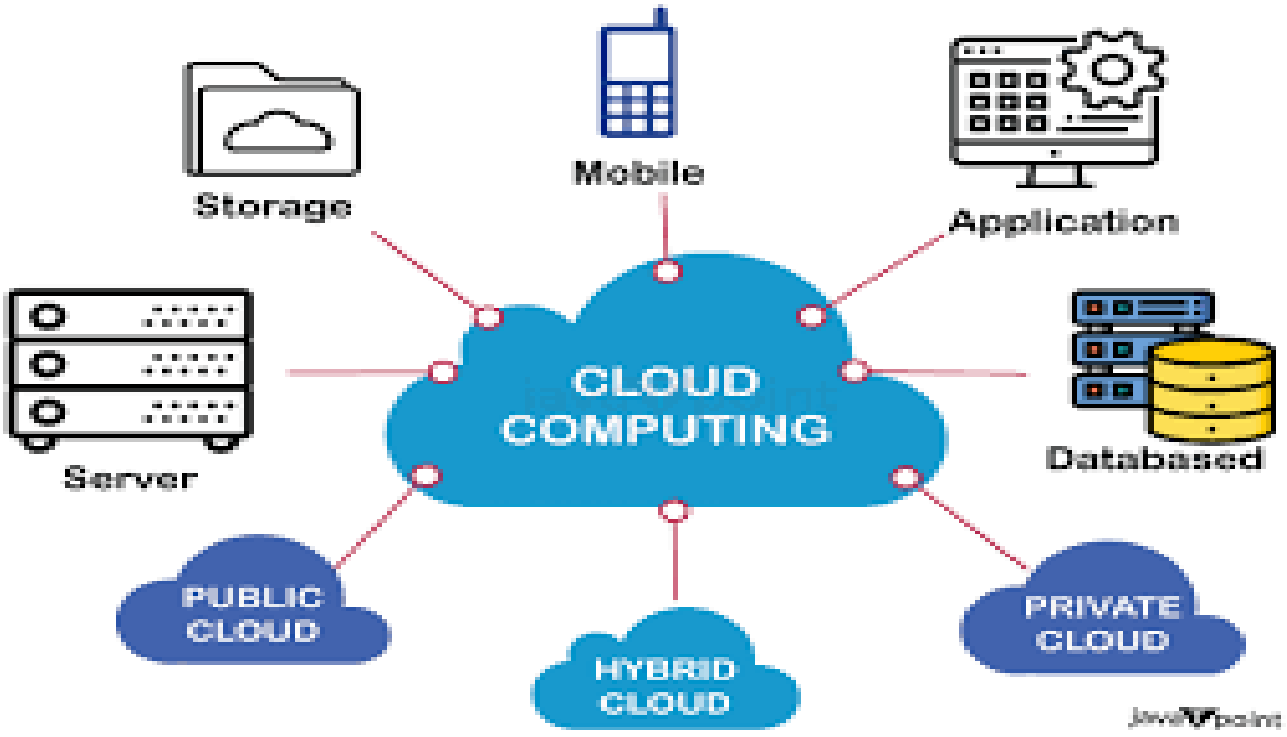

FIG 1.1 CLOUD MODELS AND SERVICES

4.PROBLEM TO SOLUTION

1. **Cloud-Based Storage**: Utilize Google Cloud Filestore for scalable, high-performance file storage with low latency.
2. **Secure User Access**: Implement Firebase Authentication for secure login and role-based access control.
3. **Web-Based Interface**: Develop an intuitive front-end for file upload, management, and retrieval using HTML, CSS, and JavaScript.
4. **API Integration**: Provide RESTful APIs for easy integration with third-party applications.
5. **Data Redundancy and Recovery**: Leverage GCP's backup, snapshots, and zonal redundancy to ensure data availability and disaster recovery.
6. **Scalable Backend**: Use Kubernetes Engine or Cloud Functions for dynamic scaling and efficient request handling.

The solution involves creating a cloud-based file storage system using **Google Cloud Filestore**, integrated with web technologies to address challenges of scalability, security, and accessibility.

1. **Storage Infrastructure**: Google Cloud Filestore offers high-performance, low-latency storage, capable of handling large volumes of data efficiently. Its scalability ensures the system adapts to growing user needs.
2. **User Authentication and Security**: Implement **Firebase Authentication** to manage user accounts securely. Role-based access controls prevent unauthorized actions, ensuring data privacy.
3. **Web Interface**: Build an intuitive, responsive front-end using HTML, CSS, and JavaScript. The interface allows users to upload, download, and manage files easily.
4. **Backend Processing**: Employ **Google Kubernetes Engine (GKE)** or **Cloud Functions** for scalable backend logic. This ensures efficient processing of user requests, even under heavy traffic.
5. **Data Protection**: Leverage GCP's redundancy, snapshots, and backup features to ensure data reliability and enable disaster recovery. This guarantees minimal downtime and secure data recovery during failures.
6. **Integration Capabilities**: Provide RESTful APIs to allow third-party applications to integrate seamlessly with the storage system, enabling versatile use cases across industries.

This solution ensures secure, scalable, and accessible file management, addressing the modern demands of data storage and retrieval.

5.IMPLEMENTATION

1. **Requirements Analysis and Planning**
    - Identify project goals: secure, scalable, and accessible file storage.
    - Define system components: front-end interface, back-end architecture, storage infrastructure, and security layers.
    - Select GCP services: Google Cloud Filestore, Kubernetes Engine, Cloud Functions, and Firebase Authentication.
2. **Setting up Google Cloud Platform**
    - Create a GCP account and project.
    - Enable necessary APIs: Filestore API, Kubernetes API, and Firebase Authentication API.
    - Allocate a budget and monitor usage using GCP Billing.
3. **Provisioning Google Cloud Filestore**
    - Set up a Filestore instance with the desired performance tier (Standard or Premium).
    - Configure the network settings to ensure secure access.
    - Mount the Filestore instance to virtual machines using NFS protocol.
4. **Back-End Development**
    - Use **Google Kubernetes Engine (GKE)** for containerized application deployment.
    - Alternatively, use **Cloud Functions** for serverless operations to handle file upload and retrieval requests.
    - Implement RESTful APIs for integrating the back-end with the front-end and external systems.
5. **Front-End Development**
    - Design a responsive web interface using HTML, CSS, and JavaScript frameworks like React or Angular.
    - Build pages for file upload, download, and management.
    - Add client-side validation for secure and optimized file handling.
6. **Authentication and Security**
    - Integrate Firebase Authentication for user login and registration.
    - Configure role-based access controls to restrict unauthorized file access.
    - Use GCP's built-in encryption for data security.
7. **Scalability and Reliability**
    - Set up auto-scaling for GKE to handle varying user loads.
    - Use load balancers to distribute traffic efficiently across the back-end services.
    - Implement redundancy and backup mechanisms for disaster recovery.
8. **Testing**
    - Perform unit, integration, and end-to-end testing to ensure system reliability.
    - Test performance under simulated high loads to validate scalability.
9. **Deployment**
    - Deploy the web application using GCP services like App Engine or Compute Engine.
    - Host the front-end using Firebase Hosting or a custom web server.
    - Monitor system performance using GCP Monitoring and Logging.
10. **Documentation and Training**

- Create detailed documentation for developers and end-users.
- Provide training to system administrators for effective maintenance.

11. **Future Enhancements**

- Plan for advanced features like file versioning, real-time collaboration, and AI-based content categorization.
- Incorporate support for additional file formats and protocols.

This implementation approach ensures a robust, scalable, and user-friendly file storage system leveraging the power of GCP.

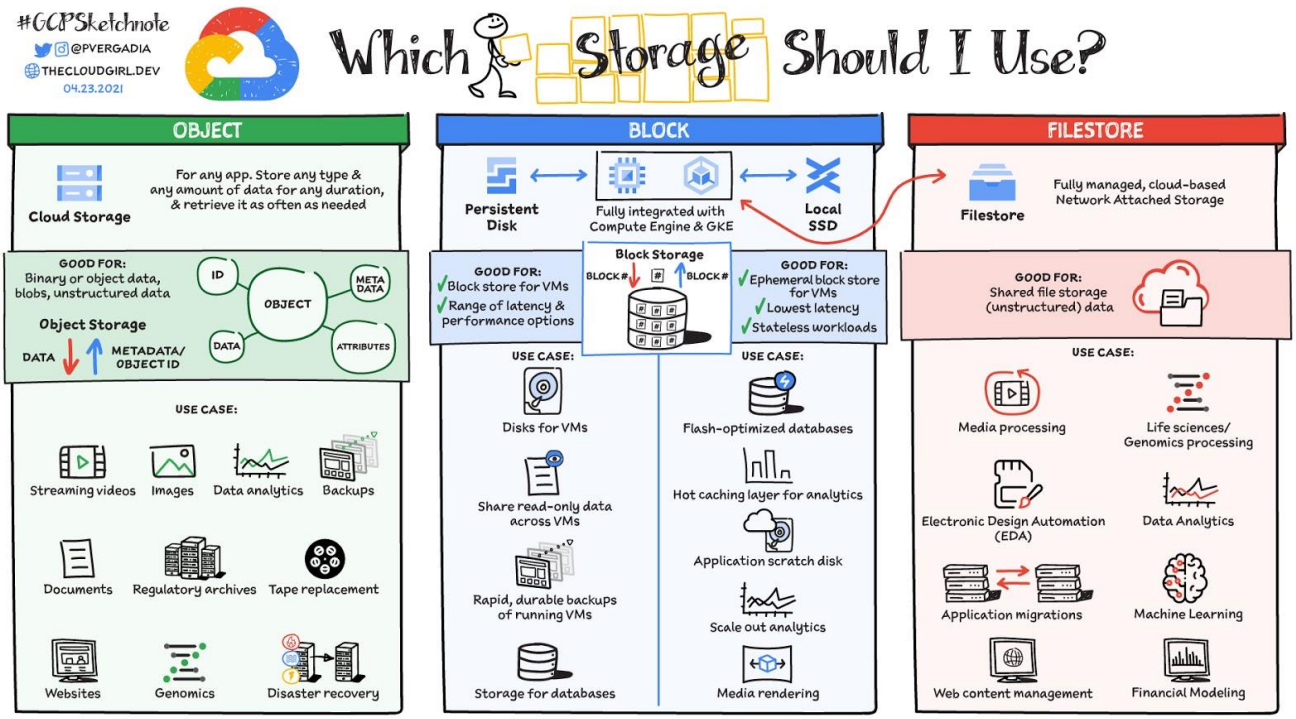

FIG 1.2 CLOUD STORAGE OPTIONS (GCP)

6.FEATURES AND FUNCTIONALITIES

User Authentication and Authorization

- **Secure Login**: Users can log in using Firebase Authentication or OAuth options like Google Sign-In.
- **Role-Based Access Control**: Administrators can assign roles to manage access levels, ensuring data security.

File Upload and Management

- **Easy Upload Interface**: Users can upload files via a drag-and-drop feature or a file picker.
- **File Organization**: Allow folder creation and file categorization for better organization.
- **Bulk Upload**: Support for multiple files uploaded simultaneously.

File Access and Sharing

- **Download Files**: Users can securely download files from the system.
- **Access Control**: Set permissions for files (private, shared, or public).
- **Link Sharing**: Generate secure shareable links with expiration options.

Real-Time Sync

- **Instant Updates**: Real-time synchronization ensures changes to files or folders reflect across all devices immediately.

Scalability and Performance

- **High Performance**: Google Cloud Filestore offers low-latency file access, even under heavy loads.
- **Auto-Scaling**: The system dynamically adjusts storage and compute resources based on user demands.

Data Security and Integrity

- **Encryption**: Data is encrypted in transit and at rest using GCP's built-in encryption features.
- **Version Control**: Maintain file versions to prevent accidental overwrites or deletions.
- **Backup and Restore**: Periodic backups ensure data recovery in case of failures.

User-Friendly Web Interface

- **Responsive Design**: The interface adapts to desktops, tablets, and mobile devices.
- **Dashboard View**: A centralized dashboard displays storage usage, recent activity, and notifications.
- **Search Functionality**: Users can quickly find files using keywords or metadata filters.

Integration Capabilities

- **API Access**: Developers can use RESTful APIs to integrate the storage system with other applications.
- **Third-Party Tools**: Support for integrating with analytics, monitoring, and automation tools.

Monitoring and Analytics

- **Activity Logs**: Track file activities such as uploads, downloads, and sharing actions.
- **Usage Metrics**: Display detailed reports on storage consumption and user activity.

Collaboration Features

- **Shared Workspaces**: Teams can collaborate by accessing shared folders.
- **Notifications**: Email or app notifications for file updates or shared access.

Support for Multiple File Types

- Compatible with various file types, including documents, images, videos, and custom formats.
- Previews for common file formats (PDF, DOCX, JPG, etc.) within the web interface.

Compliance and Reliability

- **Data Residency**: Choose storage locations to comply with regional data regulations.
- **Uptime Guarantee**: The system leverages GCP's infrastructure for high availability and reliability.

Future-Proof Enhancements

- Incorporate AI features like auto-tagging files based on content.
- Add blockchain-based file tracking for enhanced transparency.

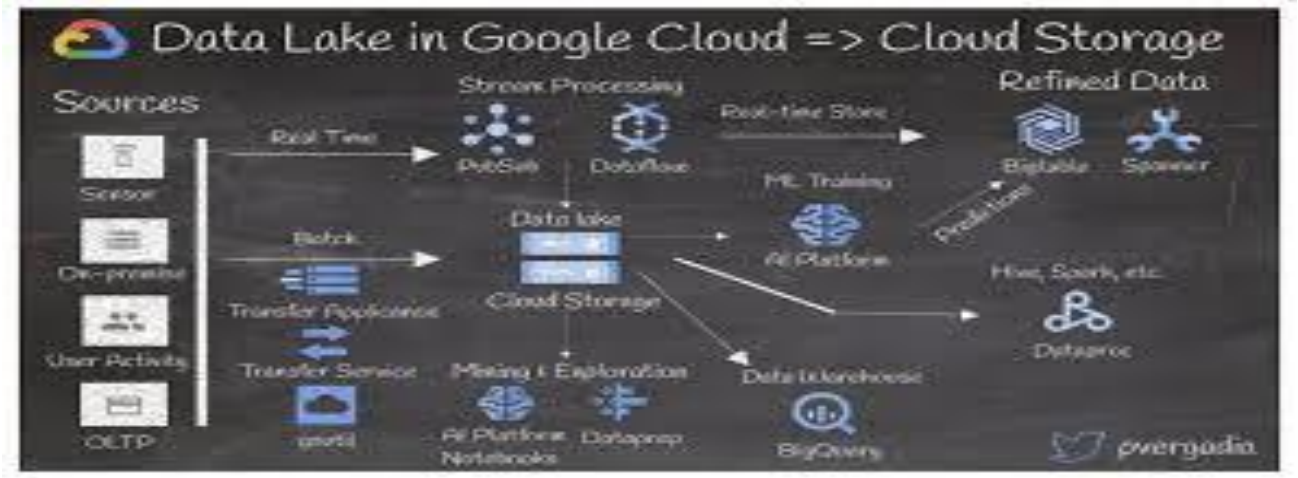

FIG 1.3 CLOUD STORAGE BUCKET

7.SYSTEM DESIGN AND ARCHITECTURE

The **GCP File Storage System** project uses a cloud-based architecture to provide secure, scalable, and high-performance file storage accessible via a web interface. The system is designed to handle modern storage challenges, offering a seamless experience for users while ensuring reliability and efficiency.

At the **frontend**, users interact with the system through a responsive web interface developed with HTML, CSS, and JavaScript (React or Angular). The user interface is intuitive, allowing users to upload, download, manage, and organize their files easily. The system also features advanced functionalities like file search, sharing, and version control. This ensures that even under heavy traffic, the user experience remains smooth and consistent.

On the **backend**, the system leverages **Google Kubernetes Engine (GKE)** or **Cloud Functions** for application logic. These services provide the flexibility to scale the application automatically based on user demand, ensuring high availability. The backend is responsible for handling requests from the frontend, processing file uploads, file retrievals, and managing access permissions. A set of **RESTful APIs** is used to facilitate communication between the frontend and backend, as well as for integration with third-party applications, making the system adaptable to various use cases.

For **file storage**, the system relies on **Google Cloud Filestore**, which offers persistent, high-performance network-attached storage. Filestore ensures fast file access with low latency, making it ideal for handling large files and providing quick retrieval times. The system also implements **role-based access control** (RBAC) to ensure only authorized users can access or modify certain files. This access control, combined with **Firebase Authentication**, provides a secure environment for managing user accounts.

Data security is a primary concern in this architecture. The system uses GCP's built-in **encryption** for both data at rest and in transit, ensuring that user data remains protected throughout the process. Regular **backups** and **snapshots** further enhance data integrity, ensuring that files can be recovered quickly in the event of data loss or system failure. Additionally, the system provides features like file versioning to protect against accidental deletions or modifications.

Scalability and reliability are critical components of the architecture. The use of **auto-scaling** ensures that the system adapts to varying levels of demand, scaling up or down automatically to maintain optimal performance. The **Google Cloud Load Balancer** distributes traffic efficiently across backend resources, preventing any single server from being overwhelmed.

For **monitoring and analytics**, the system integrates with **Google Cloud Monitoring** to track performance metrics such as storage utilization, user activity, and system health. Detailed **usage analytics** are available to administrators to monitor file access, system load, and other operational aspects, enabling proactive maintenance and troubleshooting.

The system's **integration layer** plays a crucial role in its versatility. By offering **REST APIs**, the file storage solution can easily integrate with external applications or workflows, enabling its adoption across various industries. This allows businesses to incorporate the file storage system into their existing tools and services, extending its functionality.

In conclusion, the **GCP File Storage System** is built on a modern cloud architecture that integrates cutting-edge technologies like Google Cloud Filestore, Kubernetes, Firebase Authentication, and

RESTful APIs. This architecture enables scalability, security, and high performance while providing a flexible, user-friendly solution for file management. With features like auto-scaling, role-based access control, real-time synchronization, and data backup, the system addresses the critical needs of modern businesses and individual users alike.

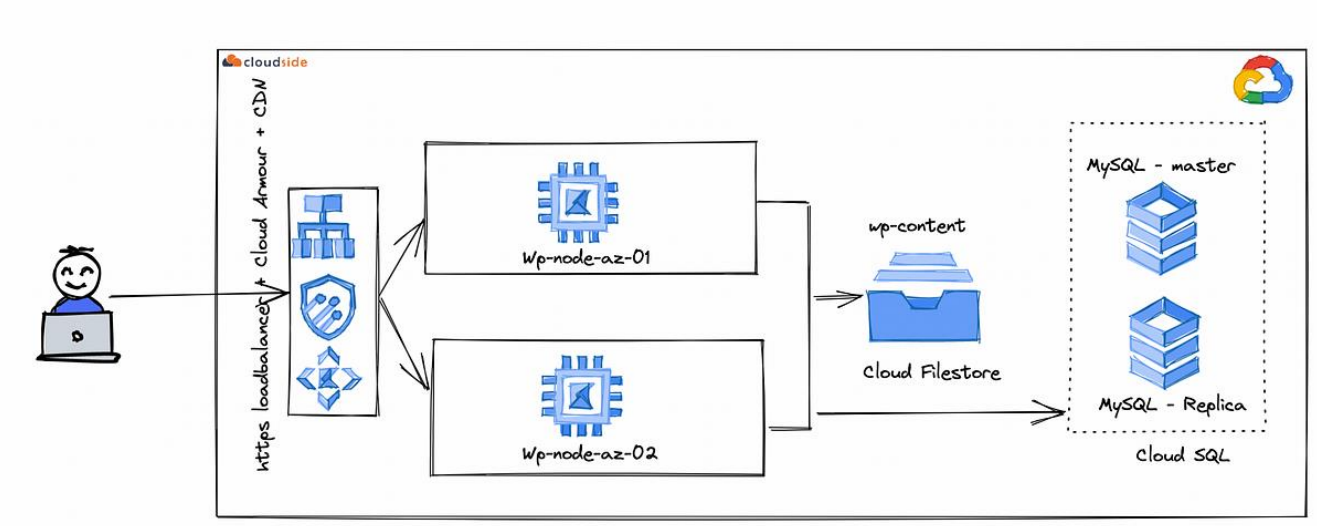

FIG 1.4 CLOUD VM INSTANCES

8.FUTURE ENHANCEMENT

1. AI-Based File Tagging: Integrate machine learning to automatically categorize and tag files based on their content, making search and organization easier.
2. Blockchain for File Integrity: Implement blockchain technology to track file modifications and ensure tamper-proof record-keeping, improving data security and transparency.
3. Real-Time Collaboration: Add collaboration features like live document editing, version control, and commenting, allowing multiple users to work on the same file in real time.
4. Extended File Format Support: Enhance support for more file formats and introduce preview options for files like CAD drawings, 3D models, or other specialized formats.
5. Cross-Platform Integration: Extend the system's capabilities to integrate seamlessly with more cloud platforms (e.g., AWS, Microsoft Azure), allowing users to sync and manage files across multiple storage providers.
6. Automated Backup and Recovery: Improve backup capabilities with automated recovery workflows that can restore systems or files to previous states based on set criteria.
7. Mobile Application: Develop mobile apps for both iOS and Android to allow users to upload, download, and manage files on the go.
8. Data Analytics & Reporting: Build advanced reporting tools for administrators to analyze storage trends, user behaviors, and potential system issues for proactive management.

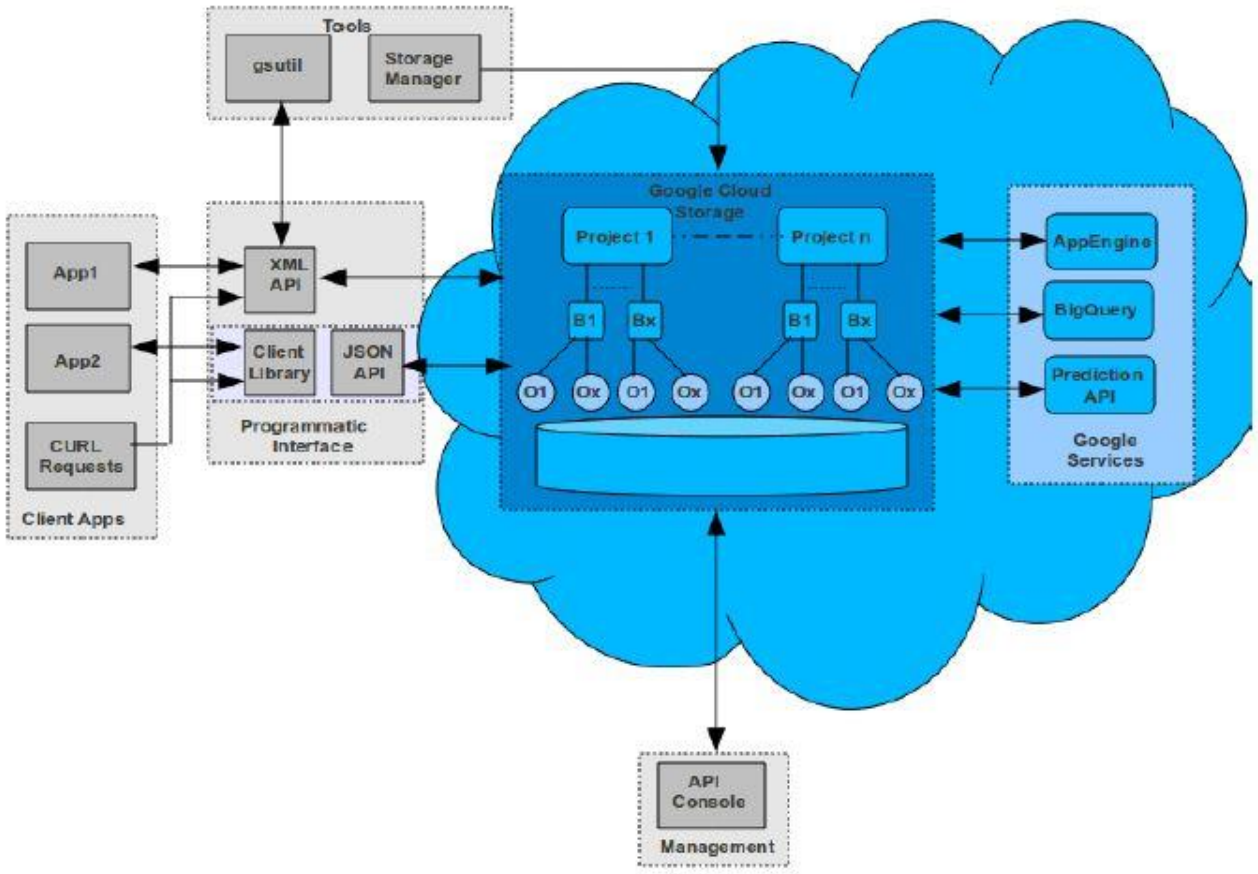

FIG 1.5 CLOUD RESOURCE HIERARCHY AND DATA FLOW

9.CONCLUSION

In conclusion, the **GCP File Storage System** project effectively leverages Google Cloud Platform's advanced services, such as **Filestore**, **Firebase Authentication**, and **Google Kubernetes Engine**, to provide a secure, scalable, and high-performance file storage solution. By incorporating modern cloud technologies, it addresses key challenges of storage accessibility, data security, and system reliability. The future enhancements, such as AI-based file tagging, real-time collaboration, and cross-platform integrations, promise to further elevate the system’s capabilities, ensuring it remains a flexible and robust solution for diverse business and personal needs.

10.REFERENCES

[1]. Vic (J. R) Winkler. Securing the cloud Computer Security Techniques and Tactics. Elsevier Inc, 2011.

[2]. Abdul Nasir Khan, M. L. Mat Kiah, Sammie U. Khan, Sajjad A. Madani. Towards secure mobile cloud computing: A survey. Future Generation Computer Systems (2012), doi:10.1016/j.future.2012.08.003.

[3]. J. Sinduja, S. Prathiba. Modified Security Frame Work for PIR cloud Computing Environment. International Journal of Computer Science and Mobile Computing-2013.

[4]. Clara Leonard. PRISM: Guardian fait de nouvelles Fromhttp://www.zdnet.fr/actualites/prism lansaargumente-le-guardian nouvellesrevelations-39791924.htm. ZDNet, Jun 28, .2013. consulted Nov 20, 2013.

[5]. Archana K Rajan, Surya Babu, “Privacy and Authenticity for Cloud Data using Attribute Based Encryption and Digital Signature” 2017 Unpublished work

[6]. J. Sánchez-García, J. M. García-Campos, D. G. Reina, S. L. Toral,F. Barrero. “On-siteDriverID: A Secure Authentication Scheme based on Spanish eID Cards for Vehicular Ad Hoc Networks.

[7]. SeDaSC: Secure Data Sharing in Clouds MazharAli, Student Member, IEEE, RevathiDhamotharan, Eraj Khan, Samee U. Khan, Senior Member, IEEE, Athanasios V. Vasilakos, Senior Member, IEEE, Keqin Li, Fellow, IEEE, and Albert Y. Zomaya, Fellow, IEEE(2017).