Workshop - Google Drive Clone

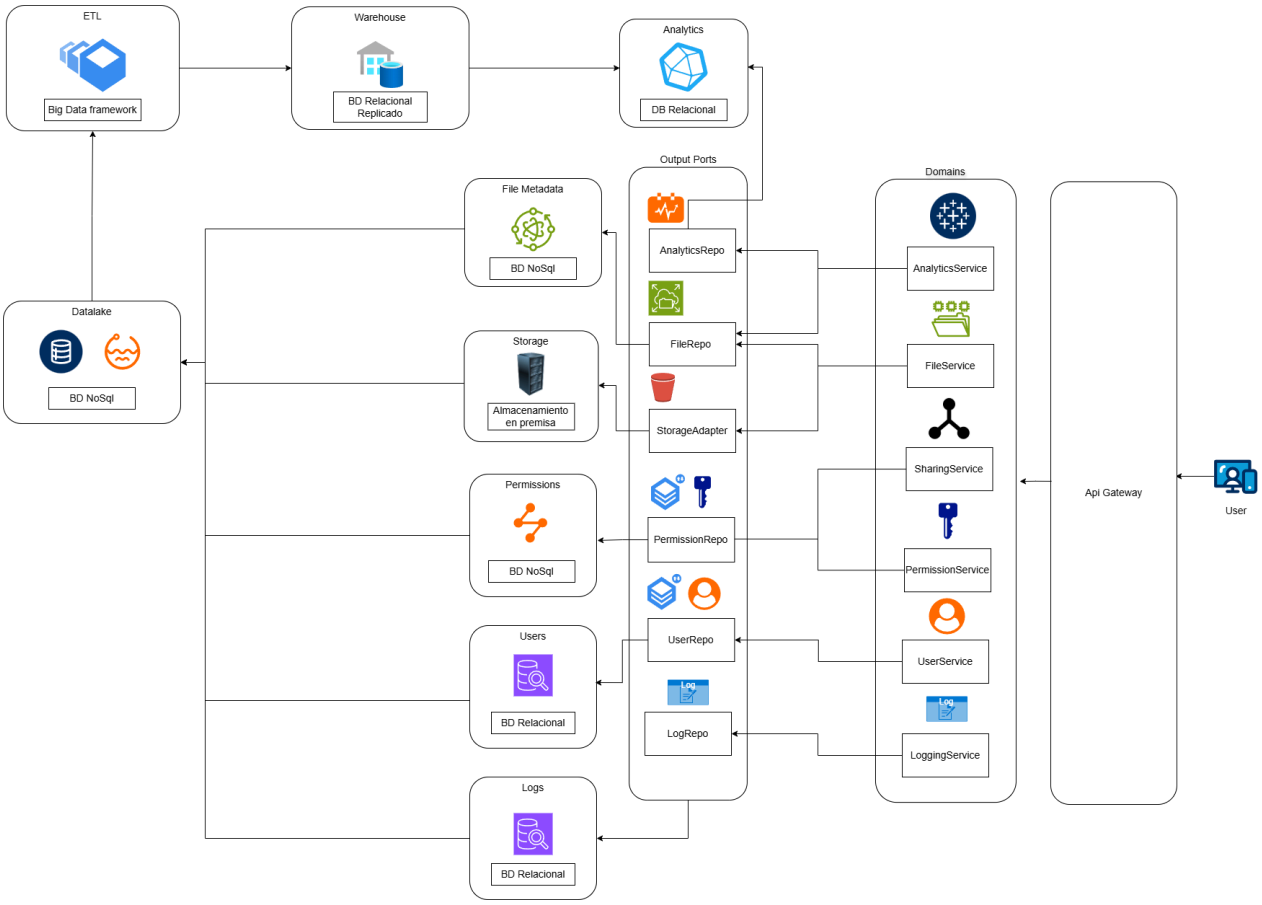
DATABASES II
Students:
Cristian David Casas Diaz - 20192020091
Dylan Alejandro Solarte Rincón - 20201020088
Professor: Carlos Andrés Sierra Virguez

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Improved Database Architecture

This system architecture represents a scalable and secure cloud-based file management platform inspired by Google Drive. The platform is designed using a hexagonal (ports and adapters) architecture, ensuring modularity, ease of testing, and maintainability.

[Link Repository Architecture V 2.0]



Roles and technologies of Each Component:

User: Web client.

API Gateway: Responsible for orchestrating communication between services and between the logic and presentation layers.

Domains: Encapsulate each service module required for the application's operation and data analysis.

- **AnalyticsService:** Responsible for managing the logic and communication with the analytics repository.
- **FileService:** Responsible for managing the logic and communication with file repositories and the storage adapter.
- **SharingService & PermissionService:** Responsible for managing the logic and communication with the permissions repositories.
- **UserService:** Responsible for managing the logic and communication with the user repository.
- **LogService:** Responsible for managing the logic and communication with the log repository.

Repositories: Responsible for connecting to and performing operations related to databases and storage.

File Metadata DB: A NoSQL database for handling file metadata. NoSQL is chosen due to the variability of file types and the specific characteristics each one may have.

Storage DB: This refers to on-premise storage for static files, aimed at providing access and downloads. An on-premise service is selected as it offers better long-term cost efficiency.

Permissions DB: Due to the complex nature of permissions, a non-relational database is the better option.

Users DB: A relational (SQL) database will be used to store user information. This type of database is chosen because the data is easily structured.

Log DB: Similar to the users case, logs are easily structured and will be stored in a relational database.

Datalake: A NoSQL database will be used to store information obtained from the data sources (the databases of each domain), acting as a datalake to store data for future cleaning and transformation.

ETL: This module will handle the data cleaning and transformation processes for storage in the data warehouse.

Warehouse: A series of relational databases, along with their replicas, will be used to store information after the ETL process and distribute it to the data marts.

Analytics: An analytics module that will function as a data mart to store information related to metrics and analytics.

System Information Requirements

User Information

Required data:

- Name, email, encrypted password
- Email verification status
- User type (standard, premium, administrator)
- Activity history (last login, changes, sharing events)

Related use cases:

- Registration and login (RF1.1 to RF1.3)
- Password recovery and profile editing (RF1.4, RF1.5)
- Access permission management (RF5.5, RF5.8)
- Two-factor authentication (RF8.4)

Files and Metadata

- **Required data:**

- File ID, name, extension, size
- File owner
- Upload date, modification date, last view
- Status (active, in trash, permanently deleted)

- File version

Related use cases:

- File upload, download, and organization (RF2.1 to RF2.6, RF3.1)
- Deleted file recovery (RF2.6)
- Quick preview (RF2.3, RF9.1)
- Large file support (User Story: *Large File Support*)

Folder Structure

Required data:

- Folder ID, name, parent-child hierarchy
- Folder owner
- Creation and modification dates

Related use cases:

- Folder creation and organization (RF3.1, RF3.2)
- Renaming and moving files/folders (User Story: *Rename and Move Files*)

Access Permissions and Sharing

Required data:

- Shared file or folder
- Users with access and their permissions (view, edit, download)
- Passwords for shared links
- Access expiration date

Related use cases:

- Sharing with users or via links (RF5.1 to RF5.8)
- User Stories: *Access Permissions*, *Shareable Links*

Activity Logs and Auditing

Required data:

- Type of activity (upload, edit, download, share)
- User who performed the action
- Date and time
- IP address or origin

Related use cases:

- Activity log display (User Story: *Activity Log*)
- Security auditing (RF8.3)

Search and Tagging Data

Required data:

- Search indexes by name, type, date, size
- Assigned tags
- Favorite files

Related use cases:

- Advanced search (RF4.1 to RF4.3)
- User Stories: *Quick Search*, *Favorite Files and Folders*

Statistics and Analytics

Required data:

- Storage usage per user
- Most accessed files
- User behavior (access frequency, most used file types)
- System performance

Related use cases:

- User Story: *Analytics Dashboard*
- RF11.1 to RF11.9

Security and Compliance

Required data:

- Successful and failed login attempts
- Malware events or violations
- Encrypted backups

Related use cases:

- Encryption, 2FA, antivirus scanning (RF8.1, RF8.2, RNF4.6)

Queries for System Information Requirements

The following queries have been designed to meet the system's various information requirements, each with a defined purpose and corresponding requirement to fulfill.

Information Requirement	Purpose	Example SQL and NoSQL Queries
User Information	Retrieve basic user profile and status	SELECT user_id, full_name, email, account_type, last_login, email_verified FROM User WHERE is_actived = TRUE;

Files and Metadata	List files uploaded by a user with metadata.	db.files.find(
		{ owner_id: 123, is_deleted: false },
		{
		file_id: 1,
		file_name: 1,
		file_type: 1,
		size_kb: 1,
		created_at: 1,
		last_modified: 1
		}
		);

Folder Structure	Retrieve a user's folder hierarchy.	db.folders.find({ owner_user_id: 123, is_deleted: false }, { folder_id: 1, folder_name: 1, parent_folder_id: 1, created_at: 1

		}).sort({ parent_folder_id: 1, created_at: 1 });

Access Permissions and Sharing	Query active access permissions for shared resources.	db.sharing.find(
		{ is_active: true, shared_with_id: 123 },
		{
		resource_id: 1,
		shared_with_id: 1,
		permission_level: 1,
		shared_at: 1,
		expire_at: 1
		}
		);

Activity Logs and Auditing	Show last 10 actions by a user	SELECT activity_type, resource_id, timestamp, ip_address FROM Activity_Log WHERE user_id = 123 ORDER BY timestamp DESC LIMIT 10;

Search and Tagging	Find files associated with a specific tag.	db.resource_tags.aggregate([
		{ $match: { tag_name: "project2025" } },
		{
		$lookup: {
		from: "files",
		localField: "resource_id",
		foreignField: "file_id",
		as: "files"
		}
		},
		{ $unwind: "$files" },
		{ $project: { file_id: "$files.file_id", file_name: "$files.file_name" } }
		]);

Statistics and Analytics	Calculate used and remaining storage for a user	SELECT u.user_id, u.used_storage, u.storage_quota, (u.storage_quota - u.used_storage) AS remaining_storage FROM User u WHERE u.user_id = 123;
Security and Compliance	Audit failed login attempts for a user	SELECT timestamp, ip_address, user_agent FROM Authentication_Log WHERE user_id = 123 AND success = FALSE ORDER BY timestamp DESC;

Improve Workshop 1

The following improvements are made according to the feedback provided, the architecture diagram was improved, and user stories for premium user and administrator roles were added, as previously suggested.

Explaining more deeply, the architecture diagram was improved including new domains and modules (Big Data related). Additionally, the storage approach was changed from the cloud to on premise, mainly because it is a better choice money-wise in the long term.

References

Amazon Web Services. (n.d.). *What is data architecture?* AWS. Retrieved May 13, 2025, from https://aws.amazon.com/es/what-is/data-architecture/

Business model canvas examples. (n.d.). *Corporate Finance Institute*. Retrieved March 17, 2025, from https://corporatefinanceinstitute.com/resources/management/business-model-canvas-examples/

IBM. (n.d.). *What is data architecture?* IBM. Retrieved May 13, 2025, from https://www.ibm.com/es-es/topics/data-architecture

UNIR. (n.d.). *Entity-relationship model*. UNIR Revista. Retrieved May 13, 2025, from https://www.unir.net/revista/ingenieria/modelo-entidad-relacion/

Workspace, G. (n.d.). *Google Drive: Share files online with secure cloud storage*. Google Workspace. Retrieved March 17, 2025, from https://workspace.google.com/intl/es-419/products/drive/