Document Management System (DMS)

Roles, Permissions, Sharing & Folder Architecture

This README documents the **complete, unified design** of a Document Management System (DMS) covering: - System roles - Folder & document hierarchy - Organization Drive vs My Drive - Access control (permissions & sharing) - Groups and cross-team sharing

This document is intended to be a **single source of truth** for developers, architects, and AI tools working on or extending this system.

---

1. Core Design Principles

1. **One hierarchy, one permission system**
2. **Roles decide WHO you are**
3. **Permissions decide WHAT you can do**
4. **Folders are the primary access boundary**
5. **Documents inherit permissions from folders by default**
6. **Sharing is the only way normal users get access**

---

2. System Roles (Very Small & Stable)

Only **three system-level roles** exist.

2.1 Super Admin

- System-wide access
- Can access all departments and folders
- Can assign admins and department owners
- **Implicit access** (not stored in permission tables)

2.2 Admin (Multi-department)

- Assigned to **one or more departments**
- Full access inside assigned departments
- Implicit access to department folders

2.3 Department Owner (Single department)

- Admin of **exactly one department**
- Same privileges as Admin, but limited scope
- Technically the same role as Admin with one department mapping

❌Folder Manager, Viewer, Editor are **NOT roles**

---

3. Folder & Document Hierarchy

A **single folders table** represents: - Organization folders - Department folders - My Drive folders - Subfolders

3.1 Folders Table

```
folders
- id
- name
- parent_id
- owner_type     ('ORG' | 'USER')
- owner_id
- department_id  (nullable)
- created_by
```

3.2 Organization Drive

- Root folder per department
- owner_type = 'ORG'
- department_id is set

3.3 My Drive

- One root folder per user
- owner_type = 'USER'
- department_id = NULL

Org vs Personal is an ownership concept, not a permission concept

---

4. Access Control (Permissions & Sharing)

A **single access control table** manages permissions for: - Folders - Documents - Users - Groups

4.1 Access Control Table

```
access_control
- id
- resource_type   ('FOLDER' | 'DOCUMENT')
- resource_id
- subject_type    ('USER' | 'GROUP')
- subject_id
- can_view
- can_create
- can_update
- can_delete
- can_share
```

4.2 Permission Meaning

Action	Meaning
can_view	View / download
can_create	Create subfolder or upload file
can_update	Rename / edit
can_delete	Delete
can_share	Share with others

---

5. Default Permission Rules

5.1 Folder Creation

When a folder is created: - **Always create one access_control row** for the creator - Creator gets all permissions

Example:

```
resource_type = 'FOLDER'
resource_id   = 20
subject_type  = 'USER'
subject_id    = creator_id
can_view      = true
can_create    = true
can_update    = true
can_delete    = true
can_share     = true
```

5.2 Implicit Access (No Table Entry)

- Super Admin → everything
- Admin / Department Owner → their department folders
- My Drive owner → their own folders

---

6. Sharing Model

6.1 How Sharing Works

- Sharing = inserting a row into `access_control`
- Sharing is allowed **only if can_share = true**

6.2 Example: Share Folder with Upload Access

```
resource_type = 'FOLDER'
resource_id   = 30
subject_type  = 'USER'
subject_id    = user_x
can_view      = true
can_create    = true
can_update    = false
can_delete    = false
can_share     = false
```

---

7. Groups & Cross-Team Sharing

7.1 Groups

```
groups
- id
- name
- department_id (nullable)
```

```
group_users
- group_id
- user_id
```

7.2 Group Permission Example

```
resource_type = 'FOLDER'
resource_id   = 50
subject_type  = 'GROUP'
subject_id    = finance_team
can_view      = true
can_create    = true
can_update    = true
can_delete    = false
can_share     = false
```

> Direct USER permission **overrides** GROUP permission

---

8. Document Permissions

8.1 Default Behavior

- Documents inherit permissions from parent folder
- No access_control row created by default

8.2 Document-Level Override

Used only when a document must be shared or restricted differently

Example:

```
resource_type = 'DOCUMENT'
resource_id   = 101
subject_type  = 'USER'
subject_id    = manager_id
can_view      = true
can_update    = true
```

---

9. Permission Evaluation Order

When a user performs an action:

1. Super Admin → allow
2. Admin / Department Owner (department folders) → allow
3. Folder owner (creator) → allow
4. Direct USER permission
5. GROUP permission
6. Parent folder inheritance
7. Deny

---

10. Key Rules to Remember

- Every folder has exactly **one creator permission row**
- Normal users **only get access via sharing**
- Roles never grant fine-grained actions
- Permissions are action-based, not role-based
- One permission table for everything

---

11. Final Summary

This DMS design provides: - Minimal roles - Maximum flexibility - Clear mental model - Enterprise-grade access control

Roles define identity. Permissions define capability.

This document can be safely used as a reference for implementation, future extension, or AI-assisted development.