# ****Go-based Institutional Repository****

**Instead of a heavy Java monolith (DSpace), we design a modular Go microservice architecture.**

## 1. ****Core Components****

### 🔹 ****Storage Layer****

* **MinIO (S3-compatible, Go-based, scalable object storage).**
* **Alternative: SeaweedFS for distributed FS.**
* **Stores digital objects: PDFs, theses, images, datasets.**

### 🔹 ****Database Layer (Metadata Catalog)****

* **PostgreSQL (preferred for complex metadata & relations).**
* **SQLite (for lightweight/single-node use).**
* **Stores:**
  + **Metadata (Dublin Core, MODS, etc.).**
  + **User/role data.**
  + **Collections/communities.**

### 🔹 ****Indexing & Search****

* **Bleve (Go) – pure Go full-text search (no external service needed).**
* **Elasticsearch (if you want enterprise-grade distributed search).**
* **Supports:**
  + **Full-text search of documents.**
  + **Faceted browsing (by author, date, subject).**
  + **Autocomplete.**

### 🔹 ****API Layer (Go Backend)****

* **Frameworks: Gin / Fiber / Echo.**
* **GraphQL (for flexible queries) or REST (for compatibility).**
* **Handles:**
  + **CRUD for metadata & collections.**
  + **File upload/download (streaming from MinIO).**
  + **Authentication (JWT/OAuth).**
  + **Protocol support: OAI-PMH, Dublin Core export.**

### 🔹 ****Frontend (UI)****

* **React or Next.js.**
* **Features:**
  + **Browse by community/collection.**
  + **Advanced search (full-text + filters).**
  + **Item detail pages (with metadata + file preview/download).**
  + **Submission forms (upload + metadata entry).**
  + **Admin dashboard (users, collections, workflows).**

### 🔹 ****Protocols****

* **OAI-PMH (Open Archives Initiative Protocol for Metadata Harvesting).**
* **Dublin Core / JSON-LD export.**
* **DOI / Handle integration for persistent identifiers.**

## 2. ****Extended Features****

### 🔹 ****Authentication & Roles****

* **OAuth2 / Keycloak integration.**
* **Roles: Admin, Submitter, Reviewer, Public User.**

### 🔹 ****Submission Workflow****

* **User uploads → metadata form → review (optional) → publish.**
* **Pluggable workflow steps (review, embargo, etc.).**

### 🔹 ****Preservation****

* **Versioning of files.**
* **Checksums for integrity.**
* **Replication (if using distributed storage).**

### 🔹 ****Interoperability****

* **REST/GraphQL APIs for integration with external systems.**
* **OAI-PMH for metadata harvesters (e.g., OpenAIRE).**
* **DOI/Handle assignment for persistent citation.**

## 3. ****High-Level Architecture****

**┌───────────────────────────────┐**

**│ Frontend │**

**│ React / Next.js (UI + Forms) │**

**└──────────────┬────────────────┘**

**│ GraphQL/REST**

**┌────────────────────────┴─────────────────────────┐**

**│ API Layer (Go) │**

**│ - Gin/Fiber/Echo │**

**│ - Auth (JWT/OAuth) │**

**│ - OAI-PMH, Dublin Core exporter │**

**│ - File Upload (to MinIO) │**

**│ - Metadata CRUD (to Postgres) │**

**│ - Search endpoints (to Bleve/Elasticsearch) │**

**└───────────┬──────────────────┬───────────────────┘**

**│ │**

**┌───────┘ ┌───────┴────────┐**

**│ │ │**

**┌───▼───┐ ┌─────▼───────┐ ┌─────▼─────┐**

**│MinIO │ │ PostgreSQL │ │ Bleve/ES │**

**│Storage│ │ Metadata DB │ │ Indexing │**

**└───────┘ └─────────────┘ └───────────┘**

## 4. ****Tech Choices****

| Layer | Go Stack Option | Alternative |
| --- | --- | --- |
| Storage | MinIO (Go) | SeaweedFS |
| Metadata DB | PostgreSQL | SQLite |
| Indexing | Bleve (Go-native) | Elasticsearch |
| API | Gin / Fiber / Echo | gRPC + REST |
| Auth | Ory Hydra, Keycloak | JWT only |
| Frontend | React / Next.js | Vue, Angular |
| Protocols | Custom Go OAI-PMH impl. | Middleware libs |

## 5. ****Example Workflow****

1. **User logs in → JWT issued.**
2. **User uploads a PDF → stored in MinIO.**
3. **Metadata (title, author, abstract, etc.) saved in PostgreSQL.**
4. **Extracted text indexed in Bleve.**
5. **Reviewer approves → item published.**
6. **Item becomes discoverable in search + available via OAI-PMH export.**