

Archivum

A high-performance, production-ready wiki system optimized for Desktop and Mobile, designed to run on a Raspberry Pi 5 (ARM64) or any AMD64 host.

Tech Stack

Layer	Technology
Backend	Go + GraphQL (gqlgen) + SQLite (modernc.org/sqlite, CGO-free)
Frontend	Vue 3 (Composition API) + Vite + Tailwind CSS
PWA	vite-plugin-pwa (offline caching, installable)
Version Control	Git (backend executes git on the storage path)
Editor	TipTap (visual) + CodeMirror (raw AsciiDoc)
Auth	LDAP + JWT with in-memory session tracking

Features

- **AsciiDoc-native** — documents stored as plain `.adoc` files, Git is the source of truth.
- **Visual ↔ Source editor** — TipTap for rich editing, CodeMirror for raw AsciiDoc, with a round-trip TypeScript bridge.
- **Git history & diff** — every save is a commit; full history and unified diffs via GraphQL.
- **Setup Wizard** — guided first-run configuration for LDAP, admin user, and storage paths.
- **PWA** — installable on mobile, fast loading, offline viewing of cached documents.
- **Multi-arch Docker** — single Dockerfile targeting ARM64 and AMD64.

Project Structure

```
Archivum/
├── backend/                                # Go application
│   ├── cmd/server/                        # Entry point
│   └── internal/
│       ├── auth/                         # LDAP + JWT
│       ├── config/                      # config.json management
│       ├── git/                         # Git operations
│       ├── graph/                       # GraphQL schema + resolvers
│       └── storage/                     # Document read/write
├── frontend/                             # Vue 3 application
│   └── src/
│       ├── bridge/                      # AsciiDoc ↔ TipTap TypeScript bridge
│       ├── components/
│       │   ├── editor/                  # VisualEditor + SourceEditor
│       │   ├── layout/                 # AppLayout + Sidebar
│       └── wizard/                     # Setup Wizard
```

```
|
|   ├── router/
|   ├── stores/
|   └── views/
└── docker/           # Dockerfile + docker-compose
    └── .vscode/      # VS Code build & deploy tasks
```

Quick Start

Prerequisites

- Docker (with access to your registry)
- Go 1.22+ (for local backend development)
- Node.js 20+ (for local frontend development)

Docker (recommended)

```
docker compose -f docker/docker-compose.yml up -d
```

Open <http://localhost:8080> and follow the Setup Wizard.

Local Development

```
# Backend
cd backend
go run ./cmd/server

# Frontend (separate terminal)
cd frontend
npm install
npm run dev
```

Configuration

Backend — `config.json`

```
{
  "storage_path": "/data/wiki",
  "db_path": "/data/archivum.db",
  "ldap": {
    "host": "ldap.example.com",
    "port": 389,
    "base_dn": "dc=example,dc=com",
    "bind_dn": "cn=reader,dc=example,dc=com",
    "bind_password": "secret"
  },
}
```

```
"jwt_secret": "change-me",  
"listen_addr": ":4000"  
}
```

Frontend — settings.json

```
{  
  "api_url": "http://localhost:4000/graphql",  
  "app_name": "Archivum",  
  "default_theme": "light"  
}
```

Docker environment variables

Variable	Default	Description
DOCKER_PATH	/config	Base path for config and data volumes
PUID	1000	File system user ID
PGID	1000	File system group ID

Production Deployment

Directory layout on the host

```
/opt/archivum/  
├─ config/  
│   └─ config.json      ← written by Setup Wizard on first access  
├─ wiki/                 ← AsciiDoc files + git repository  
└─ db/  
    └─ archivum.db       ← SQLite database (auto-created)
```

Steps

```
# 1. Create host directories  
mkdir -p /opt/archivum/{config,wiki,db}  
  
# 2. Set ownership to match PUID/PGID (default 1000:1000)  
chown -R 1000:1000 /opt/archivum  
  
# 3. Copy and edit the environment file  
cp .env.example .env  
# Edit .env – at minimum check PUID, PGID and HOST_PORT  
  
# 4. Allow the insecure local registry (if not already done)
```

Systembeskrivningsrapport

```
# Add to /etc/docker/daemon.json:
# { "insecure-registries": ["192.168.0.19:5000"] }
# Then: sudo systemctl restart docker

# 5. Pull and start
docker compose -f docker/docker-compose.prod.yml pull
docker compose -f docker/docker-compose.prod.yml up -d

# 6. View logs
docker compose -f docker/docker-compose.prod.yml logs -f
```

Open <http://<host>:8080> and complete the Setup Wizard.

Example `.env`

```
REGISTRY=192.168.0.19:5000
IMAGE_TAG=latest

ARCHIVUM_CONFIG=/opt/archivum/config
ARCHIVUM_WIKI=/opt/archivum/wiki
ARCHIVUM_DB=/opt/archivum/db

HOST_PORT=8080
PUID=1000
PGID=1000
TZ=Europe/Stockholm
```

A fully commented template is available at [.env.example](#).

Volumes at a glance

Container path	Maps to (default)	Purpose
/config	/opt/archivum/config	config.json (backend) + settings.json (frontend)
/data/wiki	/opt/archivum/wiki	AsciiDoc source files + git repo
/data/db	/opt/archivum/db	SQLite database (archivum.db)

Backup

```
# Wiki content (git repo – just rsync or tar)
tar czf archivum-wiki-$(date +%F).tar.gz /opt/archivum/wiki

# Database
cp /opt/archivum/db/archivum.db archivum-db-$(date +%F).db
```

Systembeskrivningsrapport

```
# Config
cp /opt/archivum/config/config.json archivum-config-$(date +%F).json
```

VS Code Tasks

Open the Command Palette (**Ctrl+Shift+P**) → **Tasks: Run Task:**

Task	Platform	Description
Compose: Build & Up	Windows / Linux	Build image and start the full stack locally (http://localhost:8080)
Compose: Up	Windows / Linux	Start with the last-built image
Compose: Down	Windows / Linux	Stop and remove containers
Compose: Logs	Windows / Linux	Follow container output
Docker: Build	Windows / Linux	Build registry image
Docker: Push	Windows / Linux	Push to 192.168.0.19:5000
Docker: Build & Push	Windows / Linux	Build then push
Docker: Multi-arch Build & Push	Windows / Linux	buildx push for ARM64 + AMD64
Backend: Run	Both	go run ./cmd/server
Frontend: Dev	Both	npm run dev
Frontend: Build	Both	Production Vite build

License

MIT