

WebCalibre: Application Context Document (v1.0)

1 Purpose & Vision

WebCalibre is a specialized Digital Asset Manager (DAM) designed to bridge the gap between cloud storage (OneDrive) and local metadata enrichment. It allows users to organize files while extracting and preserving deep technical metadata (EXIF for images, Document Info for PDFs) that standard file explorers ignore.

2 Core Technical Stack

- Framework: Next.js 15+ (App Router)
- Language: TypeScript
- Database: PostgreSQL via Prisma ORM
- Authentication: NextAuth.js (Auth.js)
- Styling: Material UI (MUI)
- Storage: Integrated with Microsoft OneDrive (via Microsoft Graph API)
- Processing:
 - **Images:** sharp + exif-reader
 - ****PDFs:** pdf-parse-new (v2 logic)

• 3 Application Architecture

A. Data Access Pattern (DAL)

The project follows a strict Data Access Layer pattern located in `src/data-access/`.

- **Purpose:** All Prisma queries and database interactions are isolated here.
- **Benefit:** Components and Server Actions do not talk directly to the database; they call functions from `file-nodes.ts` or `users.ts`. This provides a single point of truth for data fetching and improves security by centralizing authorization checks.

B. The Data Layer (`prisma/schema.prisma`) The app uses a **Recursive Tree Structure** for files and folders:

- **FileNode:** Represents both files and folders. Folders have a `parentId` pointing to another `FileNode`.
- **Metadata:** Stored as a JSONB field in the database, allowing for flexible, unstructured data from different file types.

C. The Extraction Engine (`src/lib/metadata-extractor.ts`) A server-side utility that:

1. Identifies file type by extension.
2. Parses the file Buffer.
3. Images: Extracts camera make, model, GPS coordinates (DMS), aperture, and ISO.

- 4. PDFs: Extracts Author, Title, Subject, Page Count, and a Text Preview.
- 5. Sanitization: Normalizes raw binary buffers and complex objects into JSON-safe strings.

D. The Transformation Layer (src/lib/transformers.ts)

Converts raw, lowercase, or inconsistent metadata into a GUI-ready object. It handles unit conversions (e.g., Shutter Speed 0.02 → 1/50s) and coordinate mapping for Google Maps.

4. Current Feature Roadmap & Status

Feature	Status	Description
Dashboard	✔ Active	Tabular view of FileNodes. Supports "Magic Fill" and file listing.
Upload & Enrich	✔ FixedUploads files to OneDrive and creates sub-folders.	
PDF Extraction	✔ Active	Uses pdf-parse-new for high-fidelity metadata.
Library	🕒 Planned	A "Discovery" view (Gallery for images, Bookshelf for PDFs)

1. Known Logic & UX Patterns

- **Magic Fill:** A core "Wow" feature where the app reads the file binary before saving to suggest metadata properties.
- **Folder Logic:** Folders can be created at the "Root" (WebCalibre folder on OneDrive) or nested within existing directories.
- **Server Actions:** All database and storage mutations are handled via Next.js Server Actions for security and speed.

1. Critical Fixes Applied

- PDF Worker Crash: Resolved by using pdf-parse-new and disabling the web-worker in the Node.js environment.
- Folder Reset Bug: Fixed in upload-view.tsx by removing the logic that cleared the targetFolderId when toggling the "New Folder" input.