**DocketWatch PDF Functionality — Project History & Design Summary**

**1. PDF Download & Storage**

* When unfiled court cases are scraped (specifically for tools like MAP, tool\_id = 26), the system automatically downloads the associated PDF documents.
* PDFs are saved to a **shared network directory**:  
  \\10.146.176.84\general\tmztools\python\final\_pdfs  
  Filenames are standardized as E<courtCaseNumber>.pdf (e.g., E750591547.pdf).

**2. PDF Tracking in Database**

* A new table, case\_documents, was created to track PDF files.
  + **Fields:** fk\_case, tool\_id, doc\_type, doc\_status, file\_path, file\_name, file\_size, date\_downloaded, date\_checked, notes, and ocr\_text.
  + Indexes were added for efficient lookups (by fk\_case, tool\_id, and uniqueness on (fk\_case, file\_name)).
* The ingestion script (final\_pdfs\_finder.py) scans the folder, detects any new PDFs, looks up the matching courtCaseNumber in the database, and inserts a new record if it’s not already present.

**3. Handling “Removed” Cases**

* Before processing, the script moves any PDFs where the linked case status is 'Removed' to a separate directory:  
  \\10.146.176.84\general\tmztools\python\final\_pdfs\_removed  
  This keeps only active PDFs in the working directory.

**4. OCR Processing and Search**

* Every new PDF is OCR-processed using Tesseract (installed at C:\Program Files\Tesseract-OCR).
  + Text is extracted using Python’s pdf2image and pytesseract, and cleaned before storage.
  + The extracted full-text is stored in the ocr\_text field (text type, for SQL Server 2016 compatibility).
* The OCR text is not shown to users but is **included in case searches**—allowing users to search for documents by keywords present only in the PDF.

**5. PDF Viewing in ColdFusion/DocketWatch UI**

* Users see a **PDF icon/link** in the Datatables UI if a PDF is available (case\_doc\_id not null).
* The link format is standardized:  
  http://tmztools.tmz.local/pdf/E<courtCaseNumber>.pdf
* IIS is configured to map /pdf/ requests to the network share for direct PDF access in the browser.

**6. Batch Processing and Automation**

* PDF discovery, download, and OCR runs as a multi-step batch process, triggered by ColdFusion (using cfexecute or a .bat file), running Python scripts in sequence:
  1. Scrape cases
  2. Download PDFs
  3. Run final\_pdfs\_finder.py for DB insert and OCR

**7. Permissions & Technical Considerations**

* Special attention was paid to **Windows service account permissions**:
  + Python packages must be installed for the service account.
  + Network share permissions must allow the service account to read/write in the target directories.
* The transition from drive-mapped (U:) paths to UNC (\\10.146.176.84\...) required codebase-wide find/replace and troubleshooting.
  + Some frameworks (Python, Node.js, ChromeDriver) are sensitive to UNC vs. mapped drive handling.

**Key Functional Outcomes**

* **Automated, reliable capture and storage of court PDFs** for tracked cases.
* **Integrated full-text OCR for document-level search**, boosting discovery of case details not present in case metadata.
* **Direct in-app access to PDFs** via links/icons, secured and mapped via IIS.
* **Seamless handling of removed/invalid cases** (no clutter, no manual cleanup).
* **Robust, index-friendly, and efficient database design** for future analytics and scaling.