**Enterprise Document Extraction: A Dual-Process Architecture**

This architecture is designed to provide a seamless and efficient experience for caseworkers by intelligently processing documents in two distinct ways: a proactive "offline" pipeline that prepares documents in advance, and a reactive "online" flow for immediate, on-demand requests.

**1. Tools and Technologies Required**

|  |  |  |
| --- | --- | --- |
| **Category** | **Service / Tool** | **Purpose** |
| **Compute** | **Azure Container Apps** | Hosts all backend microservices: the main API, the classifier/splitter, and the individual document extractors. Provides a scalable, serverless environment. |
| **Frontend** | **Azure Static Web Apps** | Hosts the React modal/popup application, providing a fast, secure, and globally distributed user interface. |
| **AI / Extraction** | **Azure AI Document Intelligence** | The core extraction engine. We will use the **Custom Classifier** for splitting and a suite of **Pre-built Models** (Paystub, ID, etc.) for extraction. |
| **Database** | **Azure SQL Database** | A fully managed, secure relational database. Chosen over NoSQL because the data is highly structured (case IDs, document metadata, status flags) and requires strong transactional integrity. **Serves as the staging database for the review process.** |
| **Storage** | **Azure Blob Storage** | Used as a temporary "staging" area for PDFs that are actively being processed before the extracted data and preview image are moved to the SQL database. |
| **Integration** | **On-Premise Scheduler** | An agent or scheduled script running within the DHS network that monitors the OnBase storage and **pushes** new documents and metadata to the Azure API via MuleSoft to initiate the offline process. |
| **Security** | **Azure Key Vault** | Securely stores all application secrets (API keys, database connection strings). |
|  | **Microsoft Entra ID (Managed Identity)** | Provides passwordless authentication for our Azure services to communicate with each other securely. |
|  | **Azure Policy** | Enforces security and compliance rules across the entire Azure environment (e.g., preventing public data access). |
| **Monitoring** | **Azure Monitor** | Centralizes all logs, performance metrics, and errors from all services for observability and troubleshooting. |

**2. The Architectural Process**

The architecture operates in two parallel, complementary flows that both feed into and read from the central Azure SQL Database.

**Flow A: The Proactive Offline Processing Pipeline**

The goal of this flow is to have as many documents as possible pre-processed and ready *before* a caseworker ever needs them.

1. **Trigger:** An **On-Premise Scheduler** (e.g., a scheduled script) runs within the DHS network, monitoring the OnBase storage for new PDF files.
2. **Initiation:** When a new PDF is found, the scheduler securely calls the main **FastAPI Backend API** (via MuleSoft), pushing the file and its associated metadata (case ID, etc.) to Azure.
3. **Staging:** The API saves the PDF to a temporary **Azure Blob Storage** container.
4. **Classification & Splitting:** The new blob triggers the **Classifier/Splitter microservice**. This service uses the **Azure AI Custom Classifier** to analyze the PDF and identify the page ranges for each individual document it contains (e.g., pages 1-2 are a pay stub, page 3 is a driver's license).
5. **Concurrent Extraction:** For each document identified, the service calls the appropriate **Extractor microservice** (e.g., the Paystub Extractor, the ID Extractor). These services run in parallel.
6. **Extraction:** Each extractor service calls the relevant **Azure AI Document Intelligence Pre-built Model** to extract the data from its assigned page range within the original PDF.
7. **Database Storage:** As each document is successfully extracted, the extractor service saves a new row in the **Azure SQL Database**. This row contains:
   * The Case ID and other metadata.
   * A Passed status flag.
   * The extracted data as a structured JSON object.
   * A small preview image of that specific document page.
8. **Cleanup:** Once all documents within the PDF have been processed, the original file is deleted from the temporary Blob Storage.

**Flow B: The Reactive Online User Interaction Flow**

This flow serves as the primary user interaction point. In most cases, it retrieves pre-processed data. It only triggers a live extraction as a **fallback mechanism** if the offline pipeline has not yet processed a newly arrived document.

1. **Initiation:** A caseworker in the "New Heights" app clicks the "Extract" button on a specific screen (e.g., the Demographics screen).
2. **Request:** The application sends a request containing the document\_id, case\_id, and screen context to the **React Modal API** (our FastAPI Backend).
3. **Database Check:** The API first queries the **Azure SQL Database** to see if this document\_id has already been processed and stored.
   * **If YES (Data Found):** The API immediately retrieves the pre-extracted data and sends it to the React modal for display. This is the primary, high-speed path.
   * **If NO (Data Not Found):** The API initiates an on-demand extraction by sending the document to the **Classifier/Splitter microservice**. The process then follows the same steps (4-8) as the offline pipeline. The React modal shows real-time progress updates to the user while this happens.
4. **Human-in-the-Loop (HITL) Review:** The React modal displays the relevant extracted documents in an editable format, with a side-by-side preview of the document image.
5. **Confirmation & Update:** The caseworker reviews, edits if necessary, and clicks "Confirm." The final, verified data is sent back to the API. The API then calls the appropriate endpoint (via MuleSoft) to map and save the data into the **respective "New Heights" application databases**. The "New Heights" screen is then refreshed, pre-filling the fields with the confirmed data for the caseworker to complete.

**3. Data Security Strategy (Handling PII)**

Handling the PII of US citizens requires a multi-layered security approach, which is a core part of this architecture.

1. **Data Encryption at Rest:** All data stored in **Azure SQL Database** and **Azure Blob Storage** is encrypted by default using Microsoft-managed keys.
2. **Data Encryption in Transit:** All communication between services is enforced to use TLS 1.2+.
3. **No Hard-coded Secrets (Azure Key Vault):** All sensitive information is stored in **Azure Key Vault** and accessed at runtime via Managed Identity.
4. **Principle of Least Privilege (Managed Identity):** Each microservice is given an identity with only the permissions it absolutely needs.
5. **Network Security:** All services are deployed within a virtual network with rules to restrict access (e.g., the SQL database only accepts connections from our specific Container Apps).
6. **Auditing and Monitoring (Azure Monitor):** All actions are logged to track who accessed what data and when. Alerts can be configured for suspicious activity.
7. **Data Retention and Cleanup:** The original PDF is deleted from Blob Storage as soon as its contents are successfully extracted and stored in the secure SQL staging database.