High-Level Design Document: Document Processing System

1. Introduction

This document outlines the high-level design for a document processing system. The system will process input documents (assumed to be in .docx format, based on the provided filename "output\extracted_docx.docx"), extract relevant information, and potentially perform other operations as specified in the full requirements (which are assumed to be contained within the "output\extracted_docx.docx" file, but are not accessible to me here). This HLD will focus on the core architecture and key modules. Further details will be elaborated in the Low-Level Design (LLD) document.

2. High-Level Architecture

The system will adopt a microservices architecture to ensure scalability, maintainability, and flexibility. The core components will include:

* **Document Ingestion Service:** Responsible for receiving input documents, validating their format, and storing them in a persistent storage system.
* **Document Extraction Service:** This service will utilize a suitable library (e.g., Apache Tika, docx4j) to extract text, metadata, and other relevant information from the input documents.
* **Data Processing Service:** This service will process the extracted data, potentially performing tasks such as cleaning, transforming, and enriching the data. Specific operations will depend on the requirements defined in the full requirements document.
* **Data Storage Service:** This service manages the persistence of both the input documents and the processed data. A suitable database (e.g., PostgreSQL, MongoDB) will be selected based on the data model and performance requirements.

* **API Gateway:** A single entry point for external clients to interact with the system. It will route requests to the appropriate microservices.
* **Monitoring and Logging Service:** This service will collect logs and metrics from all microservices, providing insights into system performance and health.

3. Network Diagram

```
+-----------------+    +-----------------+    +-----------------+    +-----------------+
|  Client App     |---->|   API Gateway   |---->| Data Processing |---->| Data Storage   |
+-----------------+    +-----------------+    +-----------------+    +-----------------+
     ^                                              |
     |                                              |
     +--------------------------------------------------------------------+
                              |
                              V
                    +-----------------+
                    | Document Ingestion|
                    |    Service     |
                    +-----------------+
                              |
                              V
                    +-----------------+
                    | Document Extraction|
                    |    Service     |
                    +-----------------+
```

```
                    |
                    V
            +-----------------+
            | Monitoring &    |
            |   Logging       |
            +-----------------+
```

4. Key Modules

* **Document Ingestion Service:**
    * Handles file upload (potentially via HTTP POST).
    * Performs file format validation (checking for .docx).
    * Stores the document in a storage system (e.g., cloud storage like AWS S3 or a file system).
    * Generates unique identifiers for each document.

* **Document Extraction Service:**
    * Receives document identifiers from the Ingestion Service.
    * Retrieves the document from storage.
    * Utilizes a library to extract text, metadata (author, date, etc.), tables, images, etc.
    * Stores extracted data in a structured format (e.g., JSON).

* **Data Processing Service:**
    * Receives extracted data from the Extraction Service.
    * Performs data cleaning, transformation, and enrichment as defined in the requirements.
    * May involve natural language processing (NLP) techniques, depending on the requirements.

* **Data Storage Service:**
    * Stores both the input documents and the processed data persistently.
    * Provides efficient retrieval mechanisms for data access.

* **API Gateway:**
    * Handles authentication and authorization.
    * Routes requests to the appropriate microservices.
    * Provides a consistent interface for clients.

* **Monitoring and Logging Service:**
    * Collects logs and metrics from all microservices using a centralized logging system (e.g., ELK stack).
    * Provides dashboards and alerts for monitoring system health and performance.

5. Formal Structure

The system will be developed using a modular, object-oriented approach. Each microservice will be developed as a separate deployable unit. We will utilize a version control system (e.g., Git) and a CI/CD pipeline to manage code and deployments. Detailed class diagrams and sequence diagrams will be provided in the LLD document.

6. Technology Stack (Tentative)

* Programming Language: Python (or Java, depending on team expertise and project requirements)
* Database: PostgreSQL (or MongoDB, depending on data model)

* Message Queue: RabbitMQ (or Kafka)

* API Gateway: Kong (or Apigee)

* Cloud Provider: AWS (or GCP, Azure)

* Document Extraction Library: Apache Tika or docx4j

This HLD provides a high-level overview of the document processing system. The LLD will provide more detailed design specifications for each module. The specific functionalities and technologies will be refined based on a complete understanding of the requirements document ("output\extracted_docx.docx").