SYSTEM DESIGN FOR THE AI APPLICATION FOR FUTURE LARGE-SCALE USAGE

Components

1. **Web Interface**:
   - **Technology**: Angular for a rich interactive application, or plain JavaScript for simpler implementations.
   - **Function**: Handles user interactions, file uploads, and displays analysis results.
2. **Application Server (FastAPI)**:
   - **Endpoint `/upload-cv/`**: Receives PDF uploads, invokes text extraction, and sends data for analysis.
   - **Authentication**: Uses JWT tokens to manage user sessions and secure API endpoints.
3. **OpenAI GPT Integration**:
   - **Function**: Receives extracted text, performs analysis based on predefined criteria, and returns insights.
   - **Security**: API keys are managed securely via environment variables.
4. **Database System**:
   - **Relational Database (e.g., PostgreSQL)**: Stores user login details, job titles, and performance metrics.
   - **Document Database (e.g., MongoDB)**: Manages dynamic attributes and document storage to quickly retrieve the historical CV analyzed results.
5. **Email Notification System**:
   - **Function**: Notifies reviewers when new CVs are processed or flags important metrics.
   - **Integration**: Can be integrated using SMTP servers or email API services like SendGrid.
6. **Load Balancer**:
   - **Function**: Distributes incoming API traffic and file uploads across multiple servers to ensure reliability and responsiveness.
   - **Technology**: AWS Elastic Load Balancing, Nginx, or similar.

Scalability and Performance Enhancements

1. **Horizontal Scaling**:
   - Add more application servers behind a load balancer to handle increased API traffic and processing demands.
2. **Database Optimization**:
   - Implement read replicas and database sharding to enhance performance and data availability.
   - Use efficient indexing strategies in both SQL and NoSQL databases to improve data retrieval times.
3. **Asynchronous Processing**:
   - Use background tasks (e.g., with Celery and RabbitMQ for FastAPI) for processing CVs, allowing the web interface to remain responsive.
4. **Microservices Architecture**:
   - Consider decomposing the application into microservices to isolate critical services like file handling, user management, and notification dispatching.
5. **Security**:
   - Implement robust security measures including HTTPS, secure headers, OAuth for APIs, and regular security audits.

Monitoring and Maintenance

- **Logging**: Use centralized logging (e.g., ELK stack or Splunk) for all services to monitor and troubleshoot.
- **Performance Monitoring**: Tools like Prometheus and Grafana monitor the system's health and performance in real-time.
- **Automated Testing and CI/CD**: Ensure robustness and reliability with continuous integration and deployment pipelines.

Documentation and Reporting

- **User Documentation**: Provide comprehensive user guides and API documentation (e.g., using Swagger UI with FastAPI).
- **System Reports**: Regular performance and security reports to stakeholders.

Future Enhancements

Innovative Approach to Salary Component Analysis in CVs

Our FastAPI CV analysis application incorporates an advanced feature called **Retrieval-Augmented Generation for Salary Components (RAG Salary Component)**. This innovative approach aims to enhance CV evaluations by comparing the salary details mentioned in CVs against industry standards. Here's how it works:

1. **RAG Salary Component Implementation**

   - **Extraction of Key Details**:
     - The system first extracts crucial information from the CV, including the **Company Name**, **Location**, and **Role** of the candidate. This extraction uses sophisticated natural language processing techniques to ensure accuracy and relevancy.
   - **Data Retrieval**:
     - To obtain reliable salary benchmarks, the application scrapes data from well-regarded sources such as levels.fyi and other salary insight platforms. These platforms are known for providing up-to-date and comprehensive salary data across various industries and roles.
   - **Median Salary Comparison**:
     - With the extracted role and company information, the application then consults the **Bureau of Labor Statistics** (BLS), which offers detailed occupational employment and wage estimates across the United States. This step allows us to compare the candidate's salary with the median salary for similar roles in the same location and industry, providing a contextual evaluation of the salary component in the CV.

2. **FAISS for Fast Searching**

   - **Efficient Similarity Search**:
     - To further refine our analysis, we utilize **FAISS (Facebook AI Similarity Search)**, a library for efficient similarity searching at large scale. FAISS is particularly adept at clustering and retrieving high-dimensional data vectors, which is essential for handling large datasets of CVs efficiently.
   - **Role and Location-Based Retrieval**:

- Using FAISS, the system can quickly find matches for specific roles aligned to particular locations within our CV database. This capability enables us to retrieve estimated salaries for similar positions at specific companies, providing a detailed comparative analysis for each CV processed.

Why This Approach?

This methodology is not just about enhancing the precision of CV analysis; it's about adding a layer of economic context to the evaluation. By integrating real-time salary data and comparing it with personal employment details, we can offer both the candidate and the employer a clear picture of where they stand in the current economic landscape. This insight can be invaluable during candidate justification for USCIS and when assessing the competitiveness of salary packages.

Benefits

- **Data-Driven Insights**: By grounding our analysis in data from trusted sources, we provide credible and contextually relevant insights.
- **Scalability**: Both RAG and FAISS are designed to handle large-scale data efficiently, making this solution robust enough to manage growing databases of CVs.
- **Enhanced User Experience**: For users, this feature means getting a realistic overview of how their salary expectations match up to market standards, which can be a crucial factor in job negotiations.

By employing these advanced techniques, our FastAPI application sets itself apart as a forward-thinking tool for market salary analysis to extract estimated income or wage for a user. By setting up your system with these components and considerations, can ensure it is robust, scalable, and secure, ready to handle large volumes of user interactions and data processing efficiently.