TITLE: ROLE-BASED ACCESS CONTROL (RBAC) DRIVEN RETRIEVAL-AUGMENTED GENERATION (RAG) SYSTEM

ABSTRACT: This project presents a secure Retrieval-Augmented Generation (RAG) system enhanced with Role-Based Access Control (RBAC). The system allows authenticated users from different organizational departments to query a knowledge base composed of department-specific PDF documents. The queries are processed by a transformer-based LLM, but only after retrieving and filtering the contextual data strictly according to the user's role. This report outlines the purpose, methodology, architecture, implementation, challenges, and outcomes of the system.

PURPOSE OF THE PROJECT: The purpose of this project is to design and implement a secure, scalable, and role-aware RAG pipeline. Traditional RAG frameworks assume equal access to the entire document store, making them unsuitable for enterprise environments where data confidentiality varies across departments. This system ensures that: 1. Users can only retrieve and generate responses from documents belonging to their department. 2. Access is validated using JWT tokens and role decoding. 3. The RAG chain enforces strict document filtering using metadata-level role constraints.

TECH STACK USED: 1. Backend Framework: FastAPI (Python) 2. Vector Database: ChromaDB with persistent local storage 3. Embedding Model: all-MiniLM-L6-v2 (HuggingFace sentence transformer) 4. LLM Model: google/flan-t5-base via HuggingFace pipeline 5. Authentication: JWT-based role extraction 6. Document Processing: PyPDFLoader and RecursiveCharacterTextSplitter 7. Frontend: Streamlit (simple UI for testing) 8. Additional Tools: ReportLab (for PDF generation), Git/GitHub for version control

SYSTEM ARCHITECTURE: The architecture consists of the following modular elements:

1. Data Ingestion Layer: - Scans department folders (e.g., Marketing, Finance, HR) - Loads raw PDF documents - Splits them into semantic chunks - Stores them into a ChromaDB vector index with metadata (role, source)

2. Authentication Layer: - Validates JWT token - Extracts username and role - Prevents unauthorized access

3. RAG Layer: - Builds a retrieval pipeline filtered by role - Retrieves top■k relevant chunks using vector similarity - Constructs a contextual prompt - Passes it to flan■t5■base for final generation

4. Frontend Query Layer: - Provides an interface for interacting with the system - Shows generated answers and document sources

5. RBAC Enforcement: - All retrievals strictly follow: filter = { "role": user.role } - Prevents cross-department access at the vectorstore level.

CHALLENGES FACED: 1. ChromaDB Filter Errors: - Complex dictionary filters led to operator expression errors. - Solution: simplified filters to {"role": role} only.

2. LCEL (LangChain Expression Language) Complexity: - Earlier implementations returned dict objects during embedding, causing crashes. - Solution: rebuilt chain using explicit RunnableMaps.

3. Pipeline Input Length Issues: - flan■t5■base has a max token limit (≈512 tokens). - Long contexts caused overflow warnings. - Solution: reduced chunk sizes and concatenation length.

4. Environment Management: - .venv size caused GitHub push failures. - Solution: updated README with .gitignore instructions and external environment setup steps.

SUMMARY OF THE PROJECT: This project demonstrates a production-ready RAG system capable of enforcing strict department■level information boundaries using RBAC. The modular design ensures: - High security through role-based filtering - High retrieval accuracy through embeddings - Maintainability through clean architecture - Extensibility for additional departments or LLM models

The result is a robust AI assistant suitable for enterprise environments where different departments require automated yet restricted access to organizational knowledge. This implementation shows how classical access■control mechanisms can be merged with generative AI systems to create secure and intelligent solutions.

CONCLUSION: The RBAC■driven RAG system successfully integrates secure authentication, controlled data retrieval, and transformer■based text generation to deliver contextual answers restricted by user roles. The outcome is a scalable, secure, and practical AI solution aligned with modern enterprise needs.