PolicyPulse — Full Technical Documentation

PolicyPulse is an AI powered HR Policy Assistant built using Retrieval Augmented Generation (RAG), LangChain, Qdrant Vector Database, FastAPI backend, Streamlit UI, and Groq LLMs. This document provides a detailed, multi page technical explanation covering the system architecture, workflow, components, and implementation.

1. System Overview

PolicyPulse enables company employees to query HR policies using natural language. The system retrieves policy relevant information from 14 official HR policy PDFs using a vector database (Qdrant), augments user questions with retrieved context, and generates accurate answers using an LLM.

Key components:

- Streamlit (Frontend UI)
- FastAPI (Backend API)
- LangChain (RAG Pipeline)
- Qdrant (Vector Search Engine)
- HuggingFace Embeddings (MiniLM-L6-v2)
- Groq LLM (llama3-8b-8192)
- Docker (Qdrant Deployment)

2. Detailed Architecture

The architecture follows a modular, service oriented structure:

1. User → Streamlit UI
2. UI → FastAPI (/ask endpoint)
3. FastAPI → RAG Engine
4. RAG Engine → Embedding Model
5. Embedding Model → Qdrant Vector DB (Similarity Search)
6. Retrieved Chunks → Prompt Builder
7. Prompt Builder → Groq LLM
8. LLM → Answer → Memory Manager
9. Memory + Answer → Streamlit UI

3. Ingestion Pipeline (Building the Knowledge Base)

The ingestion system loads all 14 policy PDFs, extracts text using LangChain loaders, splits pages into smaller chunks, generates sentence-transformer embeddings, and stores them inside Qdrant.

Steps:

- Load PDFs → PyPDFLoader
- Clean/extract text
- Split using RecursiveCharacterTextSplitter
- Embed with MiniLM-L6-v2
- Upsert into Qdrant with metadata (source, page)

Advantages:

- Fast similarity search
- Persistent vector storage
- Scalable HNSW graph index
- High retrieval accuracy for policy queries

4. Qdrant Vector Database

Qdrant is used as the policy knowledge base.

Key features leveraged:

- HNSW indexing for fast vector similarity search
- Payload storage for metadata
- Exact & approximate nearest neighbor search
- REST & gRPC support
- Docker deployment for portability

Each document chunk is stored as:

- text: policy paragraph
- metadata: {policy_name, page_number}
- vector: 384-d embedding

5. LangChain RAG Pipeline

LangChain orchestrates all steps of RAG in PolicyPulse.

Main components used:

1. Embeddings Model (HuggingFaceEmbeddings)
2. VectorStoreRetriever
3. Document Loaders
4. Text Splitters
5. Custom Prompt Templates
6. Chat Models (Groq LLM Binding)

RAG Flow:

- Embed query
- Retrieve top-k chunks
- Build context
- Construct system prompt
- Call Groq LLM

6. Prompt Engineering Strategy

The system prompt includes:

- No hallucination rule
- Only use provided context
- Bullet points only when needed
- Short-answer mode detection
- Greeting/small-talk handling
- Topic detection (WFH, leave, attendance, etc.)

This reduces hallucinations and improves accuracy.

7. Groq LLM Integration

Groq's Llama 3.x models are used with lightning-fast inference performance.

Features:

- Deterministic responses (temperature=0)
- Low latency
- High quality policy summarization
- Strong contextual understanding

Model used:

- llama3-8b-8192

8. Memory Manager

Maintains short-term conversation memory.

Functions:

- save(user, question, answer)
- view(user)
- clear(user)

Uses:

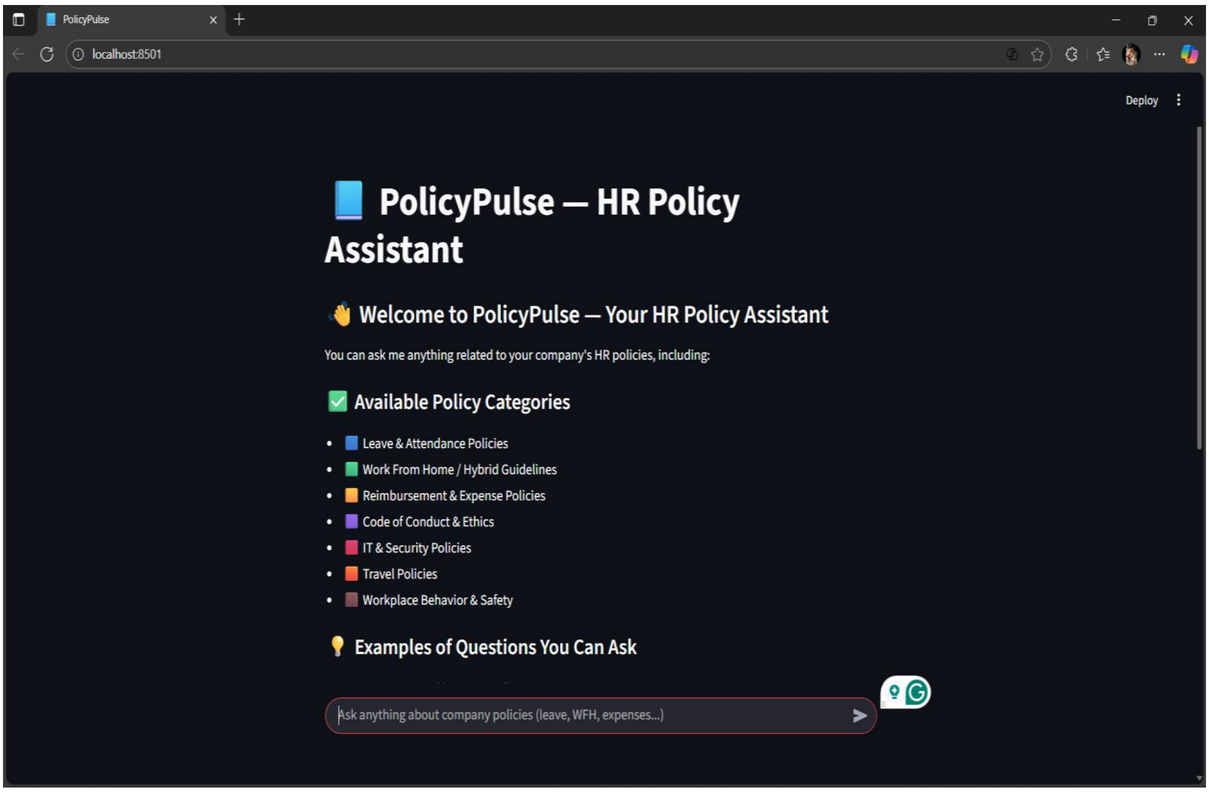
- Supports follow-up questions
- Avoids repeating context
- Prevents confusion across users

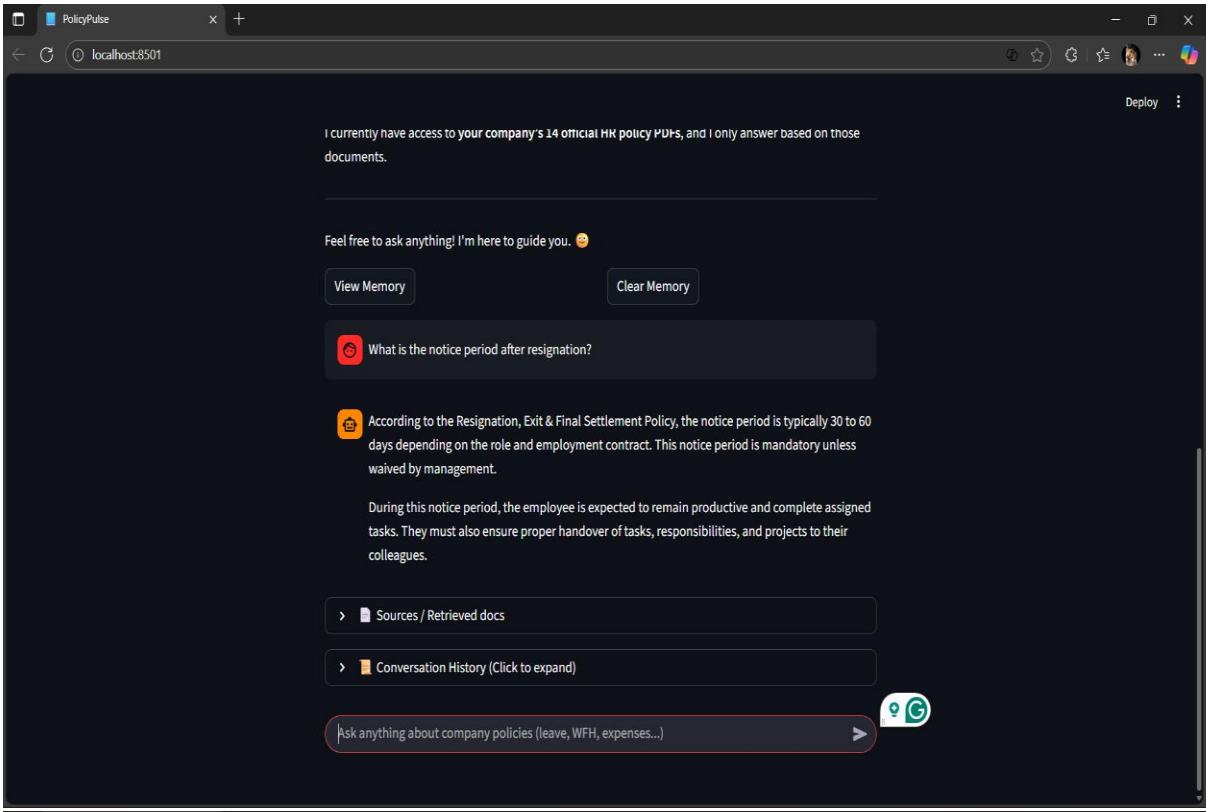
9. Full End-to-End Workflow

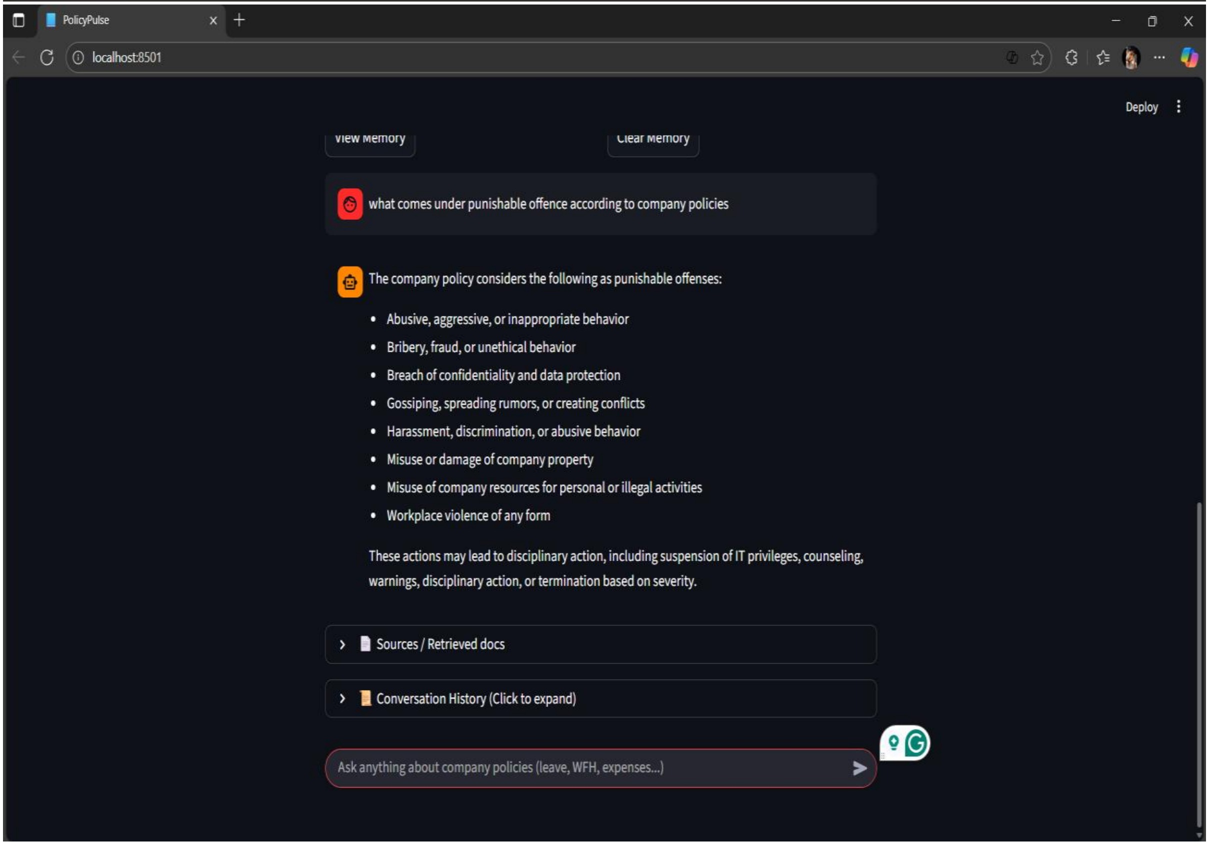
1. User asks a question in Streamlit
2. UI sends request → FastAPI
3. FastAPI calls RAG Engine
4. Query is embedded
5. Qdrant retrieves relevant policy chunks
6. Prompt is built with context
7. Groq LLM generates answer
8. Memory is updated
9. UI displays answer with typing animation

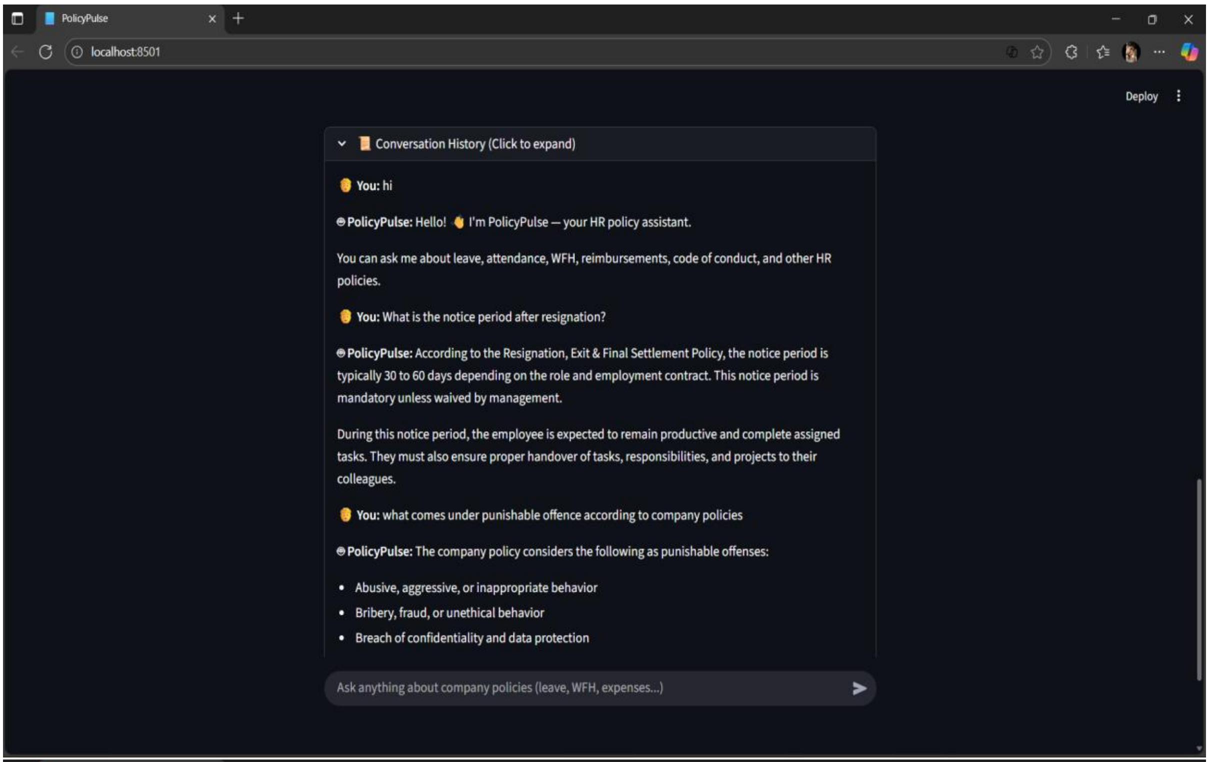
10. PolicyPulse UI Walkthrough (Screenshots)

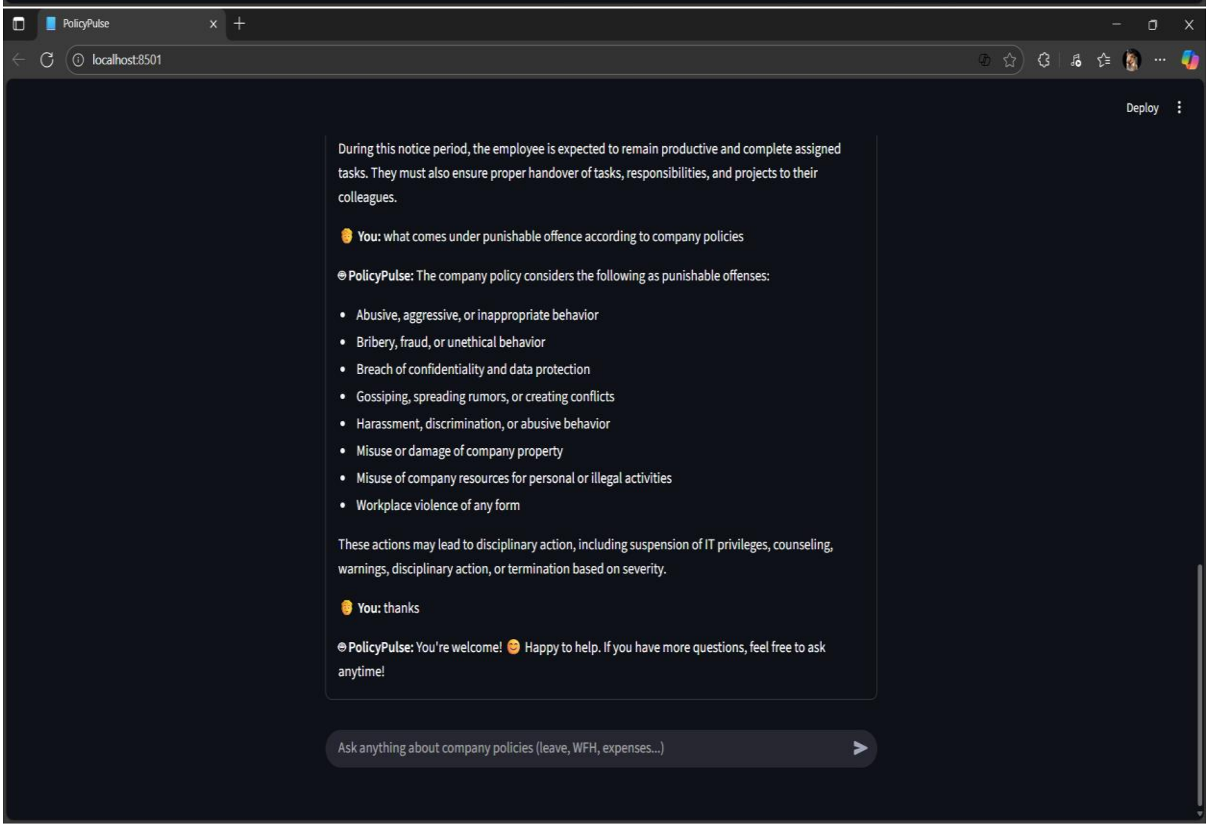
The following screenshots provide a visual overview of the PolicyPulse interface, demonstrating how employees interact with the system. These images highlight the chatbot workflow, policy responses, memory features, and the overall user experience.




PolicyPulse
localhost:8501
Deploy
I currently have access to your company's 14 official HR policy PDFs, and I only answer based on those documents.
Feel free to ask anything! I'm here to guide you.
View Memory
Clear Memory
What is the notice period after resignation?
According to the Resignation, Exit & Final Settlement Policy, the notice period is typically 30 to 60 days depending on the role and employment contract. This notice period is mandatory unless waived by management.
During this notice period, the employee is expected to remain productive and complete assigned tasks. They must also ensure proper handover of tasks, responsibilities, and projects to their colleagues.
Sources / Retrieved docs
Conversation History (Click to expand)
Ask anything about company policies (leave, WFH, expenses...)


PolicyPulse
localhost:8501
Deploy
View Memory
Clear Memory
what comes under punishable offence according to company policies
The company policy considers the following as punishable offenses:
Abusive, aggressive, or inappropriate behavior
Bribery, fraud, or unethical behavior
Breach of confidentiality and data protection
Gossiping, spreading rumors, or creating conflicts
Harassment, discrimination, or abusive behavior
Misuse or damage of company property
Misuse of company resources for personal or illegal activities
Workplace violence of any form
These actions may lead to disciplinary action, including suspension of IT privileges, counseling, warnings, disciplinary action, or termination based on severity.
Sources / Retrieved docs
Conversation History (Click to expand)
Ask anything about company policies (leave, WFH, expenses...)





10. Conclusion

PolicyPulse combines modern RAG techniques, vector search, fast LLM inference, and clear UI features to create an enterprise-ready HR assistant. This documentation provides a complete technical reference for understanding and extending the system.

📌 Author & Project Credits

Project Title: *PolicyPulse – Intelligent HR Policy Assistant using LangChain, Qdrant, and Retrieval-Augmented Generation (RAG)*
Version: 1.0
Year: 2025

👨 Author

Kavinvelavan M
LinkedIn: www.linkedin.com/in/m-kavinvelavan
GitHub: https://github.com/mkavinvelavan
Email: mkavinvelavan@outlook.com

🛠 Tech Stack Used

- LangChain
- Qdrant Vector Database
- FastAPI
- Streamlit
- Groq API (llama3-8b-8192 Model)
- Docker
- Python

🎯 Purpose of This Documentation

This documentation was created to provide a full end-to-end understanding of the PolicyPulse system architecture, workflow, retrieval pipeline, vector indexing, and RAG methodology for professional and academic use.

🔏 Ownership

This project has been fully designed, implemented, and documented by **Kavinvelavan M**
You are welcome to explore, fork, and learn from the code.