HackRx 6.0

Ideate • Co-create • Impact

BAJAJ FINSERV

Team Name : ILLVZN

Team members	Graduating year	College Name
Avinash Anish	2027	RVCE
Nihaal SP	2027	RVCE

HackRx 6.0

Ideate • Co-create • Impact

BAJAJ FINSERV

Tell us a bit about yourself

Hello, we are two friends from Information Science and Engineering from RV College of Engineering. Our github profiles: Avinash, Nihaal
Our projects: Health-Monitor, Password-Manager

Hackrx project repo

Had a lot of fun and learnt a lot in this hackathon, huge thanks to the organisers.

HackRx 6.0
Ideate • Co-create • Impact

BAJAJ FINSERV

Problem statement

Solution Overview

HackRX API is an intelligent document processing system that automatically routes between Traditional RAG and Agentic AI modes based on query complexity, delivering accurate multi-language Q&A with fast performance.

Process Flow

1. Request Processing

Client → FastAPI Gateway → Authentication → Master Agent

2. Intelligent Routing

Master Agent analyzes document type + question complexity → Routes to:

- Traditional RAG: Simple Q&A (vector search → context → LLM)
- Agentic Mode: Complex reasoning (multi-agent tool calling)

HackRx 6.0

Ideate • Co-create • Impact

3. Agentic Processing (Complex Queries)
Master Agent → Parallel Worker Agents (per question) Each Worker: Query → Tools → Summary Agent → Output Parser → Final Answer
4. Traditional RAG (Simple Queries)
Document → Vector Search → Context Assembly → LLM → Direct Answer
5. Response Assembly
All results → Master Agent → Unified JSON Response → Client Dashboard

Result: Optimal processing for both simple document Q&A and complex multi-step reasoning in a single, scalable system.

HackRx 6.0

Ideate • Co-create • Impact

BAJAJ FINSERV

Our Tech Stack:

Cloud Service Providers

LLM Providers: OpenAI, Cerebras, Gemini, Groq
Database: Supabase (PostgreSQL cloud database)
Vector Storage: In-Memory Vector Store (LangChain), Pinecone, Qdrant (as fallbacks)

Backend

Framework: FastAPI + Python 3.12+
Server: Uvicorn (ASGI)
AI/ML: LangChain, OpenAI SDK, Groq, Cerebras
Document Processing: langchain-markitdown, python-multipart, pymupdf, pymupdf4llm, easyocr, pytesseract
HTTP Client: aiohttp (async), ngrok tunnelling

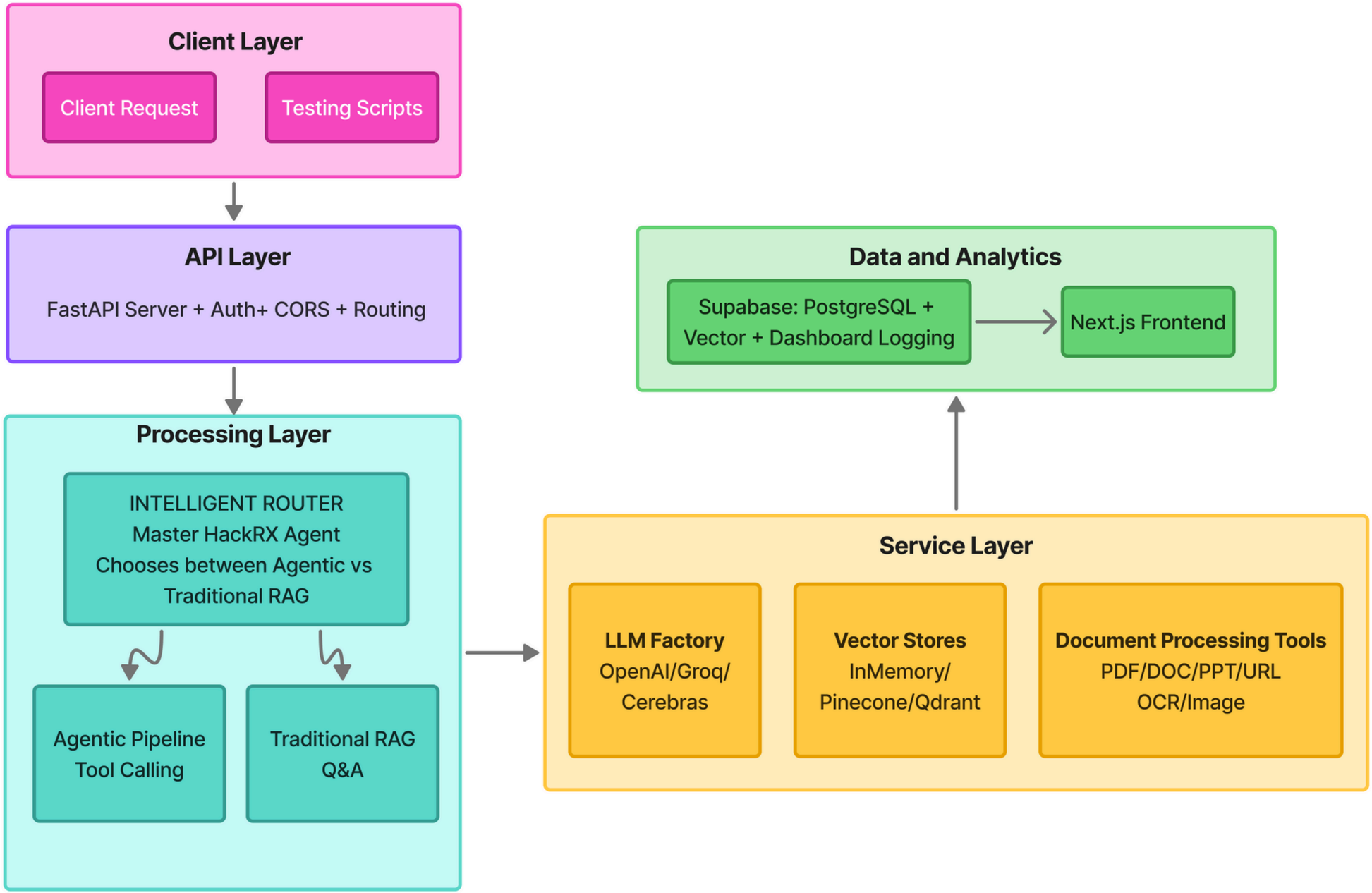
Database Architecture

Primary: Supabase PostgreSQL (cloud persistence)
Vector DB: In-Memory Vector Store (optimized for speed)
Caching: Built-in document embedding cache

Frontend

Response Dashboard: Nextjs dashboard for viewing and comparing questions and responses for better debugging
Data Source: Supabase integration for real-time data
Testing: Custom test python script (test_api.py)

Architecture



HackRx 6.0
Ideate • Co-create • Impact

BAJAJ FINSERV

Detailed description of the solution

Core Architecture

Technology Stack

- Frontend: Next.js 14 + TypeScript (Response Dashboard)
- Backend: FastAPI + Python 3.12+ with async/await patterns
- AI Layer: Hot-swappable LLM providers (OpenAI, Groq, Cerebras, Google)
- Storage: Configurable vector databases + Supabase PostgreSQL
- Patterns: Factory pattern for modularity, Singleton for VectorStore

System Components

- Master Agent: Intelligent routing and coordination
- Worker Agents: Parallel question processing with tool calling
- Traditional RAG Pipeline: Direct document Q&A for simple queries
- Tool Registry: Extensible system for document processing, vector search, and URL requests

HackRx 6.0
Ideate • Co-create • Impact

BAJAJ FINSERV

Data Flow Architecture

Request Processing

- Authentication: Bearer token validation at API gateway
- Intelligent Routing: Master Agent analyzes document type, question complexity, and context to select processing mode
- Document Processing: Multi-format parsing (PDF, Office, images) with vector embedding generation
- Dual Processing Paths:
  - Traditional RAG: Vector search → context assembly → direct LLM response
  - Agentic Mode: Multi-agent system with iterative reasoning and tool coordination

Agentic Processing Flow

- Master Agent spawns parallel Worker Agents per question
- Worker Agent performs LLM function calling with tool selection
- Summary Agent processes retrieved document context for focused summarization
- Output Parser Agent formats responses with privacy compliance and quality control
- Response Aggregation combines all worker results into unified output

HackRx 6.0
Ideate • Co-create • Impact

BAJAJ FINSERV

Performance Optimizations

- Parallel Execution: Concurrent worker agents and tool execution
- Caching Strategy: Document processing cache and vector store persistence
- Connection Pooling: HTTP session reuse and database optimization
- Async Operations: Non-blocking processing throughout the pipeline

Key Features

- Multi-Language Support
- Native Malayalam script processing alongside English
- Individual question language preservation
- Language-specific prompt optimization
- Factory Pattern Implementation
- LLM Factory: Runtime provider switching between OpenAI, Groq, Cerebras, Google
- Vector Store Factory: Configurable backends (InMemory, Pinecone, Qdrant, Supabase)

HackRx 6.0

Ideate • Co-create • Impact

BAJAJ FINSERV

- Hot-swappable Components: Environment-based configuration without code changes
- Quality Assurance
- Anti-hallucination prompts in Traditional RAG mode
- Multi-agent verification in Agentic mode
- Output parser for privacy filtering and response standardization
- Comprehensive error handling with graceful degradation
- Monitoring and Analytics
- Performance Tracking
- Tool usage analytics and provider performance monitoring
- Cache efficiency analysis and optimization opportunities

Dashboard Integration

- Next.js frontend for real-time response analysis
- Supabase logging for request/response storage and debugging
- Comprehensive error monitoring and usage pattern analysis

HackRx 6.0

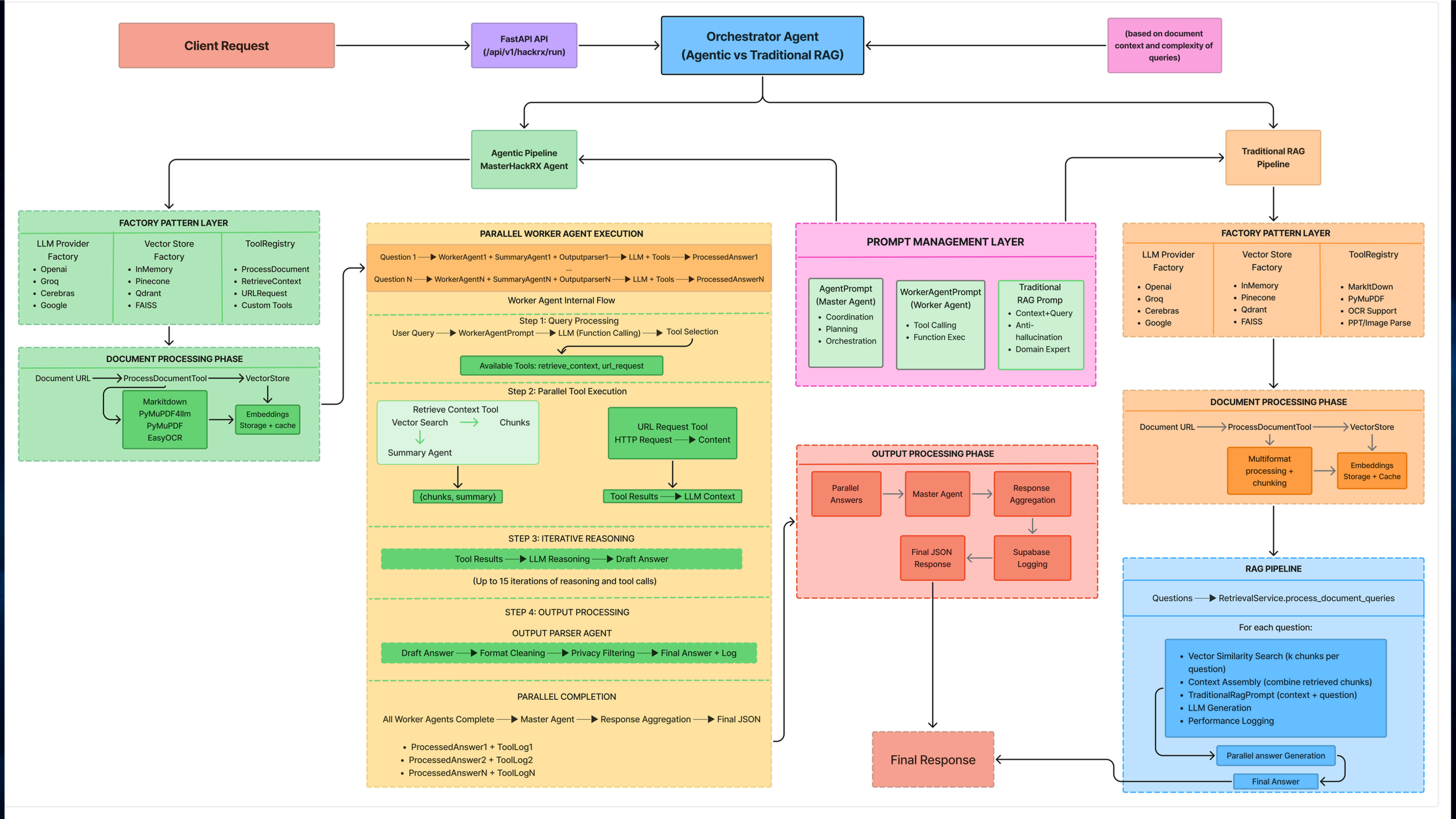
Ideate • Co-create • Impact

BAJAJ FINSERV

Data Flow Diagram is on the next slide

To view it clearer:

- Figma Link
- Drive Link


Client Request
FastAPI API (/api/v1/hackrx/run)
Orchestrator Agent (Agentic vs Traditional RAG)
(based on document context and complexity of queries)
Agentic Pipeline MasterHackRX Agent
Traditional RAG Pipeline
FACTORY PATTERN LAYER
LLM Provider Factory
Openai
Groq
Cerebras
Google
Vector Store Factory
InMemory
Pinecone
Qdrant
FAISS
ToolRegistry
ProcessDocument
RetrieveContext
URLRequest
Custom Tools
DOCUMENT PROCESSING PHASE
Document URL
ProcessDocumentTool
VectorStore
Markitdown
PyMuPDF4llm
PyMuPDF
EasyOCR
Embeddings Storage + cache
PARALLEL WORKER AGENT EXECUTION
Question 1 ▶ WorkerAgent1 + SummaryAgent1 + Outputparser1 ▶ LLM + Tools ▶ ProcessedAnswer1
...
Question N ▶ WorkerAgentN + SummaryAgentN + OutputparserN ▶ LLM + Tools ▶ ProcessedAnswerN
Worker Agent Internal Flow
Step 1: Query Processing
User Query ▶ WorkerAgentPrompt ▶ LLM (Function Calling) ▶ Tool Selection
Available Tools: retrieve_context, url_request
Step 2: Parallel Tool Execution
Retrieve Context Tool
Vector Search
Chunks
Summary Agent
{chunks, summary}
URL Request Tool
HTTP Request ▶ Content
Tool Results ▶ LLM Context
STEP 3: ITERATIVE REASONING
Tool Results ▶ LLM Reasoning ▶ Draft Answer
(Up to 15 iterations of reasoning and tool calls)
STEP 4: OUTPUT PROCESSING
OUTPUT PARSER AGENT
Draft Answer ▶ Format Cleaning ▶ Privacy Filtering ▶ Final Answer + Log
PARALLEL COMPLETION
All Worker Agents Complete ▶ Master Agent ▶ Response Aggregation ▶ Final JSON
ProcessedAnswer1 + ToolLog1
ProcessedAnswer2 + ToolLog2
ProcessedAnswerN + ToolLogN
PROMPT MANAGEMENT LAYER
AgentPrompt (Master Agent)
Coordination
Planning
Orchestration
WorkerAgentPrompt (Worker Agent)
Tool Calling
Function Exec
Traditional RAG Promp
Context+Query
Anti-hallucination
Domain Expert
OUTPUT PROCESSING PHASE
Parallel Answers
Master Agent
Response Aggregation
Supabase Logging
Final JSON Response
Final Response
FACTORY PATTERN LAYER
LLM Provider Factory
Openai
Groq
Cerebras
Google
Vector Store Factory
InMemory
Pinecone
Qdrant
FAISS
ToolRegistry
MarkItDown
PyMuPDF
OCR Support
PPT/Image Parse
DOCUMENT PROCESSING PHASE
Document URL
ProcessDocumentTool
VectorStore
Multiformat processing + chunking
Embeddings Storage + Cache
RAG PIPELINE
Questions ▶ RetrievalService.process_document_queries
For each question:
Vector Similarity Search (k chunks per question)
Context Assembly (combine retrieved chunks)
TraditionalRagPrompt (context + question)
LLM Generation
Performance Logging
Parallel answer Generation
Final Answer

Frontend Dashboard

HackRX API

Home > Dashboard

Overview

- Home
- Dashboard
- Activity

LLM QA Response Dashboard

Compare and analyze responses from your LLM question-answering system

Select Requests to Compare

Select up to 2 requests from the same document to compare their responses

19 requests https://ash-speed.hetzner.com/10GB.bin

hackrx_pdf.zip?sv=2023-01-03&spr=https&st=2025-08-04T09%3A25%3A45Z
33 requests ...Dga5%2FTwMGBO9MgLOhZS8PUjvtga2cfVk%3D

principia_newton.pdf?sv=2023-01-03&st=2025-07-28T07%3A20%3A32Z&se=
56 requests ...eUMbnUKsdEaST99F5%2FDfo7wpKg9XXF5w%3D

Success
10/8/2025, 11:26:05 am
Questions: 12
Store: unknown

Success
8/8/2025, 2:30:39 pm
Questions: 12
Store: inmemory

Success
8/8/2025, 2:23:42 pm

Super_Splendor_(Feb_2023).pdf?sv=2023-01-03&st=2025-07-21T08%3A10%3
52 requests ...sAy%2BpVKr20b3ZUo5HMz1lF9%2BJh6LQ0%3D

get-secret-token?hackTeam=2947
1 request .../utils/get-secret-token?hackTeam=2947

get-secret-token?hackTeam=2995
1 request .../utils/get-secret-token?hackTeam=2995

get-secret-token?hackTeam=3150
1 request .../utils/get-secret-token?hackTeam=3150

get-secret-token?hackTeam=1712
1 request .../utils/get-secret-token?hackTeam=1712

Happy%20Family%20Floater%20-%202024%20OICHLIP25046V062425%201.

Question 1 Comparison

Questions

Request 1

How does Newton define 'quantity of motion' and how is it distinct from 'force'?

Request 2

How does Newton define 'quantity of motion' and how is it distinct from 'force'?

Answers

27.4% similar

Request 1 — 36 unique words

Newton defines 'quantity of motion' as the measure of motion arising from the velocity and quantity of matter combined. It is essentially the product of an object 's mass and its velocity. In his framework, the motion of a whole body is the sum of the motions of all its parts, meaning that if a body has double the quantity of matter and maintains the same velocity, its quantity of motion is also double. If the velocity is doubled, the quantity of motion becomes quadruple. In contrast, 'force' is defined as the influence that causes an object to undergo a change in motion. Specifically, force is associated with the acceleration of an object and is proportional to the mass of the object being accelerated. While quantity of motion is a measure of how much motion an object has, force is what causes that motion to change. Thus, while quantity of motion is a property of a moving body, force is an external influence that can alter that motion.

Request 2 — 41 unique words

Newton defines the "quantity of motion" as the measure arising from the velocity and the quantity of matter combined. Specifically, the motion of a whole body is the sum of the motions of all its parts; for example, a body double in quantity moving at the same velocity has double the quantity of motion, and with twice the velocity, it has quadruple the quantity of motion. In contrast, "force" (or more precisely, the "motive force") is related to the accelerative force acting on the matter. While quantity of motion depends on velocity and mass, force arises from the accelerative force applied to the mass, representing the power to change motion. Newton describes the innate force of matter, "vis insita," as the resistance of a body to changes in its state of rest or uniform motion. Thus, quantity of motion is a measure of how much motion a body has (mass times velocity), whereas force is the cause that changes that motion by accelerating the body.

Common Keywords (34)

newton defines the measure motion arising from velocity and quantity matter combined. its whole body sum motions all that has +14 more

Context Documents Comparison

HackRx 6.0
Ideate • Co-create • Impact

BAJAJ FINSERV

So, how if your solution different?

Intelligent Dual-Mode Processing
Unique: Automatic routing between Traditional RAG and Agentic AI based on query complexity - most solutions use only one approach

Multi-Agent Architecture
Differentiator: 3-agent system per question (Worker + Summary + Output Parser) ensures quality control and privacy compliance

Hot-Swappable Infrastructure
Advantage: Factory pattern allows runtime switching between LLM providers and vector databases without code changes

HackRx 6.0

Ideate • Co-create • Impact

Core USP: Only solution combining intelligent routing, multi-agent quality control, and production-ready performance in a single, extensible architecture.

Production-Ready Performance
Edge: Fast latency with parallel processing, async architecture, and comprehensive caching strategies

Quality Assurance Pipeline
Differentiator: Built-in anti-hallucination measures, PII filtering, and multi-stage verification unlike basic RAG implementations

Developer Experience
Advantage: Complete monitoring dashboard, real-time analytics, and extensible tool

HackRx 6.0
Ideate • Co-create • Impact

BAJAJ FINSERV

Future possible enhancements

AI & Models
- Claude/Llama integration, fine-tuned domain models, multi-modal processing

Performance
- Microservices architecture, Redis caching, GPU acceleration, auto-scaling

Features
- Conversation memory, batch processing, mobile apps, voice interface

Enterprise
- Role-based access, audit logging, on-premise deployment, enterprise integrations

Analytics
- Advanced metrics, A/B testing, cost optimization, data export APIs

HackRx 6.0
Ideate • Co-create • Impact

Risks/Challenges/Dependencies

Risks

- LLM Provider Outages: External API dependencies (OpenAI, Groq, Cerebras)
- Rate Limiting: API quotas during high traffic
- Memory Constraints: In-memory vector storage limits
- Multi-language Accuracy: Malayalam processing quality varies by provider

Dependencies

- External Services: LLM providers, Supabase, cloud vector databases
- Internet Connectivity: Required for all cloud services
- API Keys: Service disruption if expired/invalid
- Complete LLM Provider Failure: All external AI services down
- Supabase Outage: Analytics and logging unavailable

Mitigation

- Multi-provider failover setup
- Local InMemory fallbacks
- Circuit breakers for graceful degradation

HackRx 6.0

Ideate • Co-create • Impact

BAJAJ FINSERV

Acceptance Criteria Coverage

- Document Processing

Multi-format support (PDF, Office, images), intelligent parsing, vector embedding

- AI-Powered Q&A

Dual processing modes (Traditional RAG + Agentic), intelligent routing, multi-language support

- Performance

Sub-10 second latency, async architecture, parallel processing

- Quality Assurance

Anti-hallucination prompts, privacy compliance, error handling

- Production Ready

RESTful API, authentication, monitoring dashboard, real-time analytics