Equipe 22

HACKATON BI PIPELINE

05 - 06 - 07 Nov. 2025 - ESILV - IBM

NAME OF TRACK:
Education/Automation

PRESENTED BY:
BARREZ Lucas, JEGATHEESWARAN Jeyatheepan, QUÉNEL Maxim, REDON Guillaume and RENOIR Théo

Agenda

Context

Current solution

Manual search over more than 400 Q&As

→ Inefficient, slow and lacks of context

Goal

Creating an intelligent assistant to improve user support.

→ Faster, personalised

Project Objectives

User Interface

Clone Help Center and integrate an intelligent conversational agent

Self-learn Dataset

If the information is not found, create a new line in the db with the question and the answer an admin will add.

Intelligent Help Center

Use a RAG to enhance answers : understand context to propose the most relevant answer.

Improve what exists

Rather than replacing a functioning tool, focus on improving it.

Key Features

Contextual understanding of student questions.
AI-generated responses enhanced with retrieved data.
Seamless human validation when AI has no answer.
Continuous improvement through new entries.
Integrated into existing PLV Help Center database

Dataset Overview

Dataset containing 491 rows with columns title, questions, answers, topics...

Continuously enriched as human-validated answers are added to the database

Structured and cleaned for efficient retrieval within RAG pipeline

System Architecture

Frontend Next.js

Chat interface in the student portal + admin page.

Retriever

Fetches relevent entries from database, using cosinus similarity.

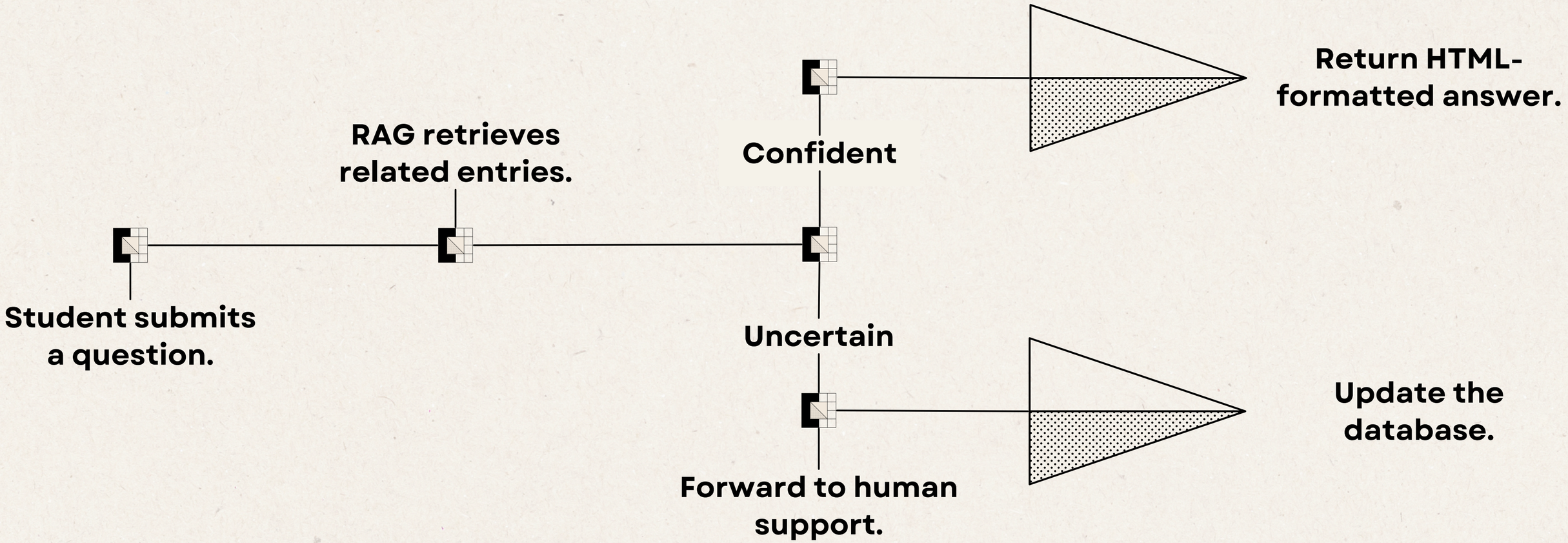
Generator (LLM)

Produces or enriches answers using retrieved data.

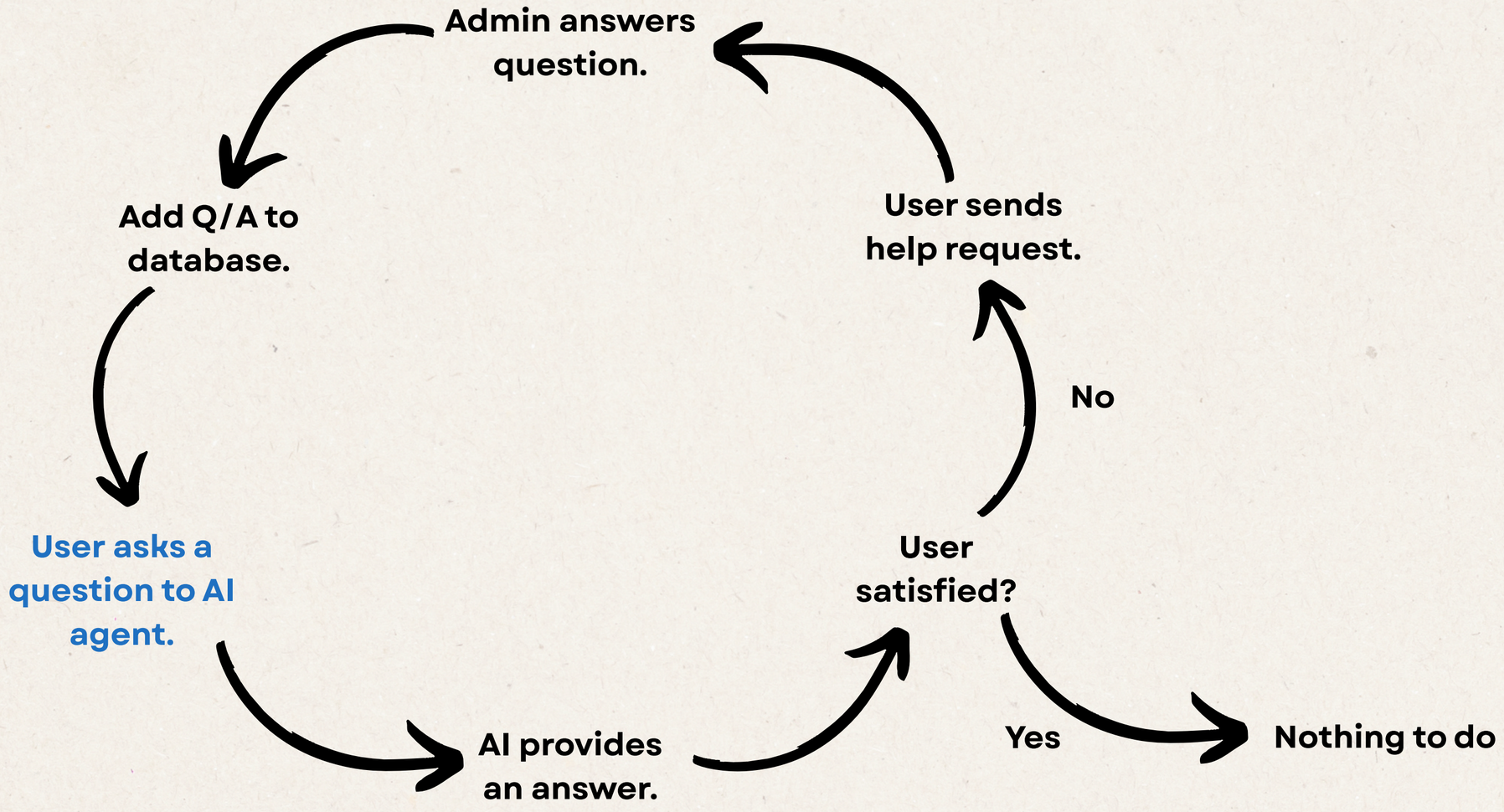
Database ChromaDB

Stores Q&A and new verified answers.

Data Flow

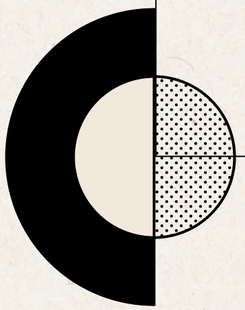

Student submits a question.
RAG retrieves related entries.
Confident
Return HTML-formatted answer.
Uncertain
Forward to human support.
Update the database.

Human-in-the-Loop Training



Technologies Used

AI Core : Retrieval-Augmented Generation (RAG)	Version Control : GitHub for Collaboration and documentation
Frontend : Nextjs	Backend : Python-based API (watsonx) logic for RAG pipeline
Database : ChromaDB	Security : Token authentification



Collaboration on GitHub

WatsonX's Problems	Rename creation error.png to creation_error.png
certification	Add files via upload
source	Connect frontend and backend
.gitignore	update gitignore
README.md	Add files via upload
README_backend.md	Add files via upload
README_frontend.md	Add files via upload
demo_video.mp4	Add demo_video.mp4
pitchdeck.pdf	Add initial pitch deck PDF

```
hackathon_IBM_DIA_E22/
├── source/
│   ├── backend/
│   │   ├── app/
│   │   │   ├── tools/
│   │   │   │   ├── document_loader.py      # JSON to LangChain Document converter
│   │   │   │   ├── rag_system.py           # Chroma vector DB & retrieval
│   │   │   │   └── IBMWatsonxChat.py       # IBM watsonx.ai API wrapper
│   │   │   └── process.py                  # RAG chain assembly
│   │   └── main.py                         # Entry point & orchestration
│   ├── frontend/
│   │   └── help-center/                    # React frontend
│   └── database/
│       ├── samples/
│       │   └── clean-json-file.json        # Source FAQ data
│       └── prod/                           # Persisted Chroma database
└── .env                                    # Environment variables (not in repo)
```

Link to the repository

Thank you !

Any question ?