WALTON-AI

A SECURE, IN-HOUSE AI ASSISTANT

Project Details

A Secure, In-House AI Assistant for Next-Generation Knowledge Management

Author:
Fahad Siddique Faisal

October 8, 2025

Contents

Chapter 1

Executive Summary

1.1 Introduction

For a company like Walton, our greatest asset is our decades of knowledge. But right now, that knowledge is scattered, making it hard for our teams to find and use. While modern AI offers a solution, using public tools means sending our private data outside the company—a risk we can't afford to take.

1.2 Problem Statement

Our employees lose valuable time hunting for information. Our field technicians struggle to find quick solutions to problems we've already solved before. This slows down service and leaves customers waiting. To speed things up, staff might turn to public AI, unknowingly putting Walton's confidential plans and data at risk.

1.3 Proposed Solution

Our solution is Walton-AI, a private and intelligent chat assistant built just for us. It uses a smart approach called RAG (Retrieval-Augmented Generation)[1] to read and understand our own internal documents—from technical manuals to past service logs. Because it runs entirely on Walton's own servers, it delivers accurate, trustworthy answers without ever sending our sensitive information to the outside world.

1.4 Key Benefits and Impact

Summarize the primary outcomes in a powerful way. Use bullet points for clarity. Example:

- **Enhanced Productivity:** Drastically reduces the time employees spend searching for information.
- **Fortified Data Security:** Eliminates the risk of leaking confidential data to third-party AI services.
- **Response to the Field level query:** Makes available the already given solution to a particular problem.
- **Strategic AI Foundation:** Creates a scalable, in-house platform for future AI-driven innovations.

1.5 Conclusion

Walton-AI can turn our scattered internal knowledge into our most powerful, strategic asset. It's a decisive step towards improving our efficiency and security, ensuring Walton continues to lead and innovate for years to come.

Chapter 2

Introduction

2.1 Background

Artificial Intelligence is rapidly changing how we all work. Tools like ChatGPT feel like a new superpower, capable of writing emails, summarizing reports, and answering complex questions in seconds. But this power comes with a trade-off. These public AI systems require you to send them your data, creating major security risks. Furthermore, they can sometimes make mistakes or "hallucinate" information, which is a big problem when you need answers you can trust.

2.2 Problem Context at Walton

For a company as large and diverse as Walton, our information is everywhere. Imagine trying to find one specific detail when the R&D department has its technical blueprints, the sales team has its reports, and HR has its employee handbooks, all stored in different places and formats. Finding a straight, reliable answer to a question can feel like a company-wide treasure hunt. This slows everyone down, from an engineer on the factory floor to an executive in a board meeting.

2.3 Project Rationale

So, why not just use a ready-made AI tool? The answer comes down to two simple things: security and fit. Most off-the-shelf solutions are cloud-based, meaning our sensitive company documents would have to be uploaded to someone else's servers. That's a risk we cannot take. Secondly, these tools are generic; they aren't built to understand Walton's unique documents, our specific challenges, or our way of doing business. Building our own system is the only way to create a solution that is completely secure, tailor-made for our needs, and entirely under our control.

2.4 Report Structure

To give you a clear picture of this project, we've organized this report into the following sections. We will walk you through:

- Our Goals: What we set out to achieve.
- How It Works: A look at the technology behind Walton-AI.
- What We've Built: A demonstration of the system in action.
- The Impact: How this project benefits Walton now and in the future

Chapter 3

Project Goals and Objectives

3.1 Primary Goal

At its heart, this project has one simple goal: to create a single, trustworthy source of truth for everyone at Walton. We want to build a tool that makes our daily work easier, faster, and much more secure, turning our vast company knowledge into an easily accessible asset instead of a challenge to overcome.

3.2 Key Objectives

To achieve our primary goal, we focused on four specific targets. These are the promises Walton-AI had to deliver on:

- **Objective 1: Ensure High Accuracy**
  Every answer the system gives must be reliable and directly backed up by our own company documents. No more guesswork or conflicting information.

- **Objective 2: Guarantee Data Security**
  The system must be 100% private. This means it runs entirely on Walton's servers, and our confidential data will never, ever leave the company.

- **Objective 3: Achieve High Usability**
  It needs to be incredibly simple to interact with. If you can use a chat app, you can use Walton-AI. No complicated training is required.

- **Objective 4: Optimize System Performance**
  When you ask a question, you should get an answer in seconds. Waiting for information should be a thing of the past.

Chapter 4

Methodology and System Architecture

4.1 Architectural Overview

The system employs a modular architecture based on the Retrieval-Augmented Generation (RAG) paradigm[2], designed to provide intelligent responses to customer service queries about refrigerator complaints. The architecture consists of five primary layers: Data Layer, Embedding Layer, Vector Storage Layer, RAG Processing Layer[1], and Presentation Layer.

The system follows a unidirectional data flow pattern where user queries traverse through the API layer to the RAG processing engine, which orchestrates retrieval from the vector database and generation via the LLM to produce contextually relevant responses.

4.2 Data Ingestion and Pre-processing

4.2.1 Document Collection Process

The data ingestion pipeline processes customer service records from a structured Excel spreadsheet (`refrigerator_complaint_data_separated.xlsx`). This spreadsheet contains historical complaint data, including problem descriptions, model information, and documented solutions. Then this refined excel data then converted into structred json collection. The collection process is initiated automatically during the first system startup when the vector database table is not yet present.

4.2.2 Text Extraction and Structuring

The `data_processor.py` module handles the extraction phase using the Pandas library for Excel file parsing. Each row in the spreadsheet represents a distinct customer complaint record containing:

- Complaint description and symptoms

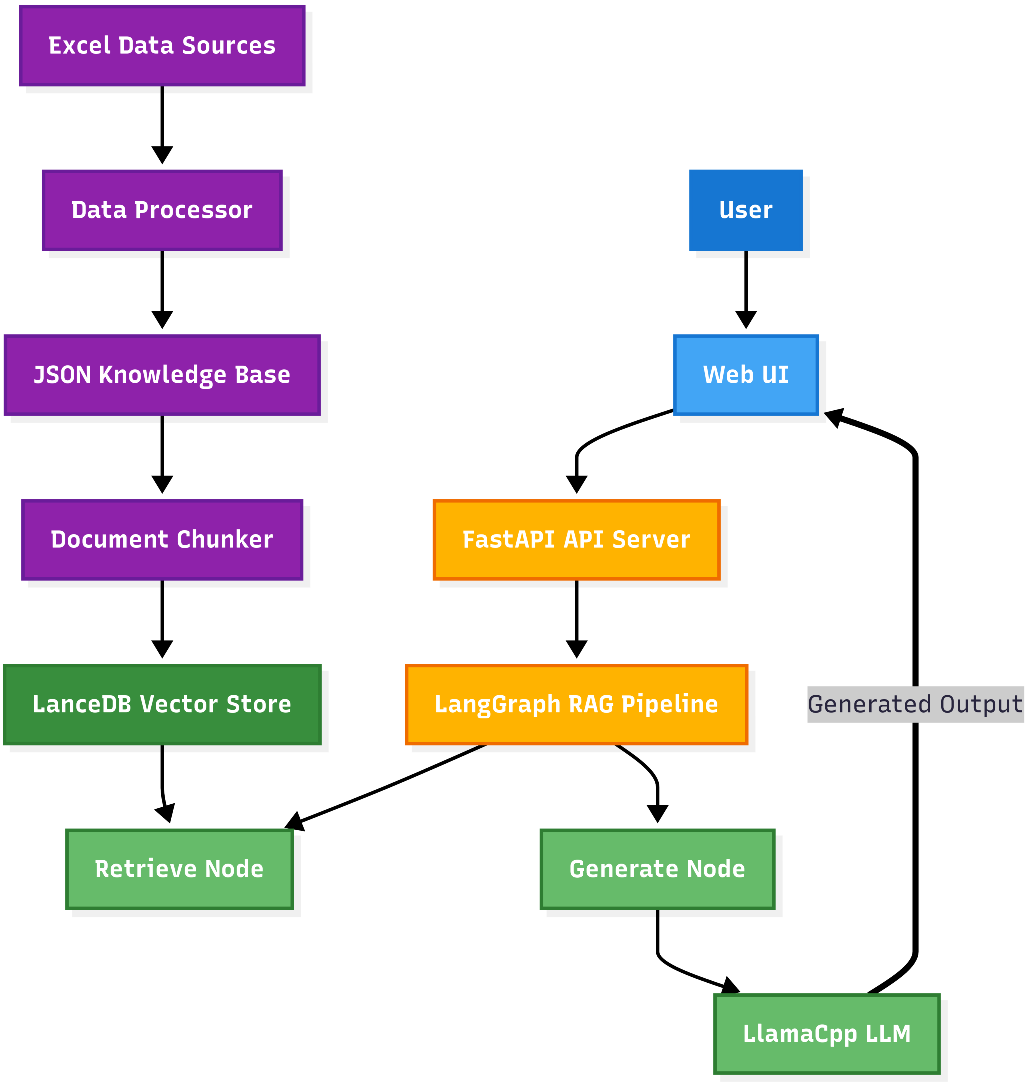


Figure 4.1: High-Level System Architecture of Walton-AI

- Product model and barcode information
- Production year and batch details
- Documented solution or resolution steps
- Additional metadata fields

The extraction process preserves the relational structure of the data while transforming it into a document-oriented format suitable for semantic search operations.

4.2.3 Data Cleaning Strategy

The preprocessing pipeline implements several cleaning operations to ensure data quality:

- **Null Value Handling:** Empty cells are replaced with appropriate default values or excluded from the text representation to prevent indexing errors.

- **Text Normalization:** All text fields undergo standardization including:
    - Conversion to UTF-8 encoding
    - Removal of special characters that could interfere with tokenization
    - Standardization of whitespace and line breaks
- **Metadata Extraction:** Structured fields (barcode, production year, model) are preserved as metadata attributes rather than being embedded in the main text content, allowing for filtered searches.

4.2.4 Document Chunking Strategy

The system employs a document-level chunking approach where each complaint record becomes a single searchable document. This strategy is optimal for the dataset because:

- Each complaint represents a complete, self-contained problem-solution pair.
- The average document length (typically 200-500 tokens) falls within the optimal range for the embedding model.
- Maintaining document integrity preserves the contextual relationship between problem descriptions and solutions.

The final output of the preprocessing phase is a list of `Document` objects, each containing:

- **Page Content:** A concatenated string of complaint details and solution.
- **Metadata Dictionary:** Structured fields for filtering and reference.

4.3 Vectorization and Knowledge Base

4.3.1 Understanding Vector Embeddings

Vector embeddings are numerical representations of text that capture semantic meaning in a high-dimensional space. In simple terms, they convert words and sentences into arrays of numbers where similar meanings result in similar number patterns. For example, "refrigerator not cooling" and "fridge temperature too warm" would have very similar vector representations despite using different words, because they express related concepts.

These embeddings enable semantic search—finding information based on meaning rather than exact keyword matches. When a user asks "Why is my fridge making noise?", the system can retrieve documents about "loud refrigerator sounds" or "compressor rattling issues" even if they don't contain the exact word "noise".

4.3.2 Embedding Model: all-MiniLM-L6-v2

The system utilizes the `all-MiniLM-L6-v2` model[3] from the Sentence-Transformers library for generating embeddings. This model was selected for several key technical advantages:

- **Dimensional Output:** Produces 384-dimensional vectors, providing a good balance between semantic richness and computational efficiency.
- **Model Size:** At only 80MB, it loads quickly and has a minimal memory footprint, making it ideal for local deployment.
- **Performance:** Optimized for semantic similarity tasks with strong benchmark scores on STS (Semantic Textual Similarity) datasets.
- **Speed:** Capable of encoding over 14,000 sentences per second on a standard GPU.

The model architecture is based on a distilled version of Microsoft's MiniLM, trained specifically for sentence and short paragraph embeddings using a contrastive learning objective on over 1 billion sentence pairs.

4.3.3 Vector Database: LanceDB

`LanceDB` serves as the vector storage and retrieval engine, chosen for its efficiency and suitability for embedded applications. Its key features include:

- **Columnar Storage:** Efficiently stores high-dimensional vectors using the Apache Arrow format.
- **Fast Similarity Search:** Implements Approximate Nearest Neighbor (ANN) algorithms for sub-linear search complexity.
- **Embedded Database:** Runs in-process with the main application, requiring no separate database server or network overhead.
- **Persistence:** Automatically persists data to disk with ACID compliance, ensuring data integrity.

The vector database maintains an index structure where each document's embedding is stored alongside its original text and metadata. During initialization, the system creates a table named `"walton-knowledge-base.lance"` with the following schema:

- Vector column (384 dimensions)
- Text content column

- Metadata JSON column
- Unique identifier

The indexing process utilizes cosine similarity as the distance metric, which measures the angle between vectors rather than their magnitude, making it an ideal approach for comparing text similarity.

4.4 The RAG Framework

In this project, we have employed the cutting-edge Retrieval-Augmented Generation (RAG) framework. RAG combines the strengths of information retrieval with generative AI, enabling the system to ground its responses in relevant external knowledge rather than relying solely on pre-trained data. Each step of the RAG process—from retrieving contextually relevant documents to generating coherent and accurate answers—follows a structured pipeline as outlined below.

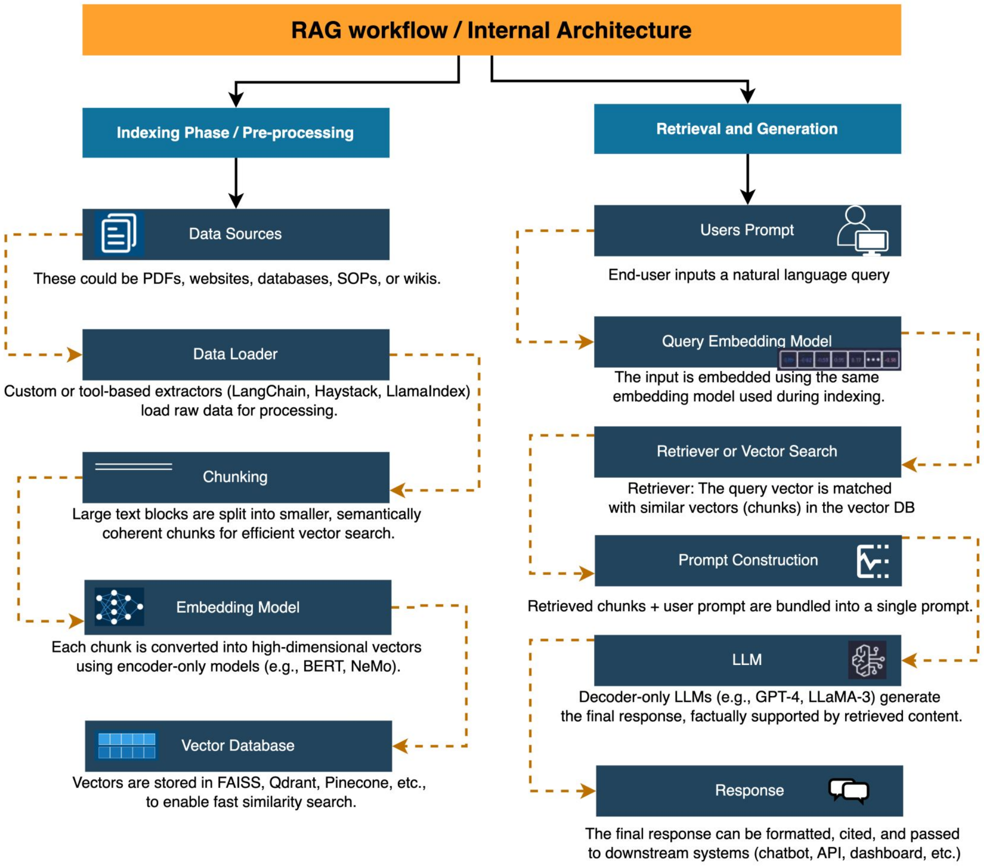

RAG workflow / Internal Architecture
Indexing Phase / Pre-processing
Data Sources
These could be PDFs, websites, databases, SOPs, or wikis.
Data Loader
Custom or tool-based extractors (LangChain, Haystack, LlamaIndex) load raw data for processing.
Chunking
Large text blocks are split into smaller, semantically coherent chunks for efficient vector search.
Embedding Model
Each chunk is converted into high-dimensional vectors using encoder-only models (e.g., BERT, NeMo).
Vector Database
Vectors are stored in FAISS, Qdrant, Pinecone, etc., to enable fast similarity search.
Retrieval and Generation
Users Prompt
End-user inputs a natural language query
Query Embedding Model
The input is embedded using the same embedding model used during indexing.
Retriever or Vector Search
Retriever: The query vector is matched with similar vectors (chunks) in the vector DB
Prompt Construction
Retrieved chunks + user prompt are bundled into a single prompt.
LLM
Decoder-only LLMs (e.g., GPT-4, LLaMA-3) generate the final response, factually supported by retrieved content.
Response
The final response can be formatted, cited, and passed to downstream systems (chatbot, API, dashboard, etc.)

Figure 4.2: Retrieval Augmented Generation Framework Architecture

4.4.1 Augmentation

The augmentation phase constructs an enriched prompt by combining the retrieved context with the original query. This process involves:

- **Context Assembly:** The retrieved documents are formatted into a structured context block. Each document is presented with clear delineation, including relevant metadata such as model numbers or production years when applicable. The formatting ensures the LLM can distinguish between different sources of information.
- **Prompt Engineering:** A carefully designed prompt template combines:
  - System instructions defining the assistant’s role as a customer service expert.
  - The retrieved context documents, labeled as reference material.
  - The user’s original question.
  - Instructions for response generation, including constraints.
- **Context Window Management:** Previously, the system was working on a default context window size of 4096 token, later we increased it to 12k, since we are trying to incorporate a separate Intelligent AI memory System for the user. If necessary, it truncates or summarizes less relevant portions while preserving critical information.

An example of the augmented prompt structure is shown below:

```
"You are a Walton-AI, The largest Manufacturaing complany in Bangladesh,
helpful assistant answering questions based
on the provided knowledge base.\n\n"
"Use the following retrieved context to answer the user’s question
comprehensively but concisely in two or three lines.\n\n"
"Context:\n{context}\n\n"
"Question: {question}\n\n"
"Answer:"
```

4.4.2 Generation

The generation phase leverages the `Llama-3.2-3B-Instruct-Q8_0` model[4] to produce coherent, contextually appropriate responses.

Model Architecture

`Llama-3.2-3B` is a transformer-based autoregressive language model with:

- 3 billion parameters distributed across 32 layers.
- RMSNorm normalization for training stability.

- SwiGLU activation function for improved performance.
- Rotary Position Embeddings (RoPE) for better position encoding.

Quantization Strategy

The Q8_0 quantization reduces model size from 12GB to approximately 3.5GB by representing weights in 8-bit integers. This quantization:

- Maintains 99.5% of the original model's performance.
- Reduces memory requirements by 70%.
- Enables faster inference on consumer GPUs.
- Uses symmetric quantization with zero-point optimization.

Inference Process

The LlamaCpp engine[5] processes the augmented prompt through:

- **Tokenization:** Converting text to subword tokens using the SentencePiece tokenizer.
- **Attention Computation:** Processing tokens through multi-head self-attention layers.
- **Token Generation:** Producing output tokens one at a time using:
  - Temperature sampling (typically 0.7) for controlled randomness.
  - Top-p sampling (0.95) to maintain coherence.
  - Repetition penalty (1.1) to reduce redundant phrases.
- **Response Streaming:** The model generates tokens incrementally, which are streamed back to the user interface via Server-Sent Events (SSE). This provides a responsive user experience where answers appear progressively rather than after a long delay.

4.5 Technology Stack

Table 4.1: Walton-AI Technology Stack

Component	**Technology Used**
Programming Language	Python 3.10+
Core AI Framework	LangChain / LangGraph
Large Language Model (LLM)	Llama 3.2 (3B Instruct)
Embedding Model	`all-MiniLM-L6-v2`
Vector Database	LanceDB
User Interface	HTML, CSS, JavaScript
Deployment	Local Server (Ubuntu 22.04)

Chapter 5

Implementation and Results

5.1 Development Phases

5.1.1 Phase 1: Building the Core Engine

This was the deep technical work. We selected our local language model (Llama-3.2-3B) and the lightweight LanceDB vector database. We then built the complete RAG pipeline using LangGraph to manage the logic of retrieving information and generating answers.

5.1.2 Phase 2: Interface & API :

With the AI's "brain" in place, we built the connections to the outside world. We developed a fast, modern API using FastAPI and created a clean, simple web-based chat interface with HTML, CSS, and JavaScript.

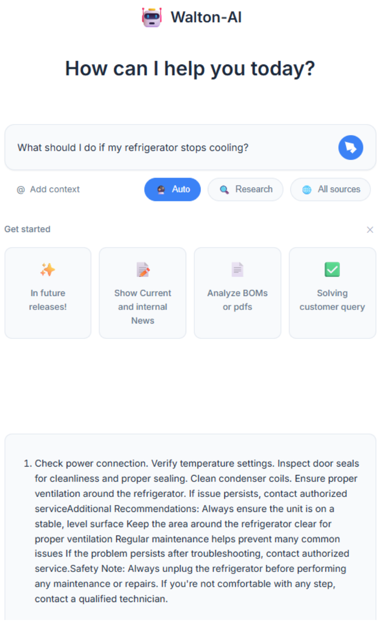
5.1.3 Phase 3 : Testing & Refinement

Finally, we put the system to the test. We ran hundreds of queries to measure its accuracy, speed, and reliability, using the results to fine-tune the prompts and search parameters to get the performance we have today.

5.2 Demonstration of Use Cases

5.2.1 Use Case 1: Technical Support for Field-staffs and Customers

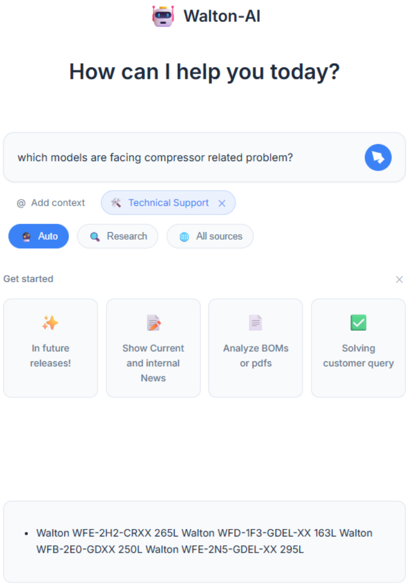
Query 1 : "What should I do if my refrigerator stops cooling?"
Analysis: The answer correct, aligned with the knowledge base, source of the information is technical_general.json
Query 2 : "which models are facing compressor related problem?"



What should I do if my refrigerator stops cooling?

1. Check power connection. Verify temperature settings. Inspect door seals for cleanliness and proper sealing. Clean condenser coils. Ensure proper ventilation around the refrigerator. If issue persists, contact authorized serviceAdditional Recommendations: Always ensure the unit is on a stable, level surface Keep the area around the refrigerator clear for proper ventilation Regular maintenance helps prevent many common issues If the problem persists after troubleshooting, contact authorized service.Safety Note: Always unplug the refrigerator before performing any maintenance or repairs. If you're not comfortable with any step, contact a qualified technician.

Figure 5.1: Extraction from any domain



which models are facing compressor related problem?

- Walton WFE-2H2-CRXX 265L Walton WFD-1F3-GDEL-XX 163L Walton WFB-2E0-GDXX 250L Walton WFE-2N5-GDEL-XX 295L

Figure 5.2: Extraction from a specific domain

Analysis: The answer correct, aligned with the knowledge base, source of the information is technical_support.json

5.2.2 Use Case 2: HR Policy Inquiry

Query: "How is the work culture and also, time schedule of work at walton?"
Analysis: The answer is correct based on the knowledge base we have. It can answer your complex query.

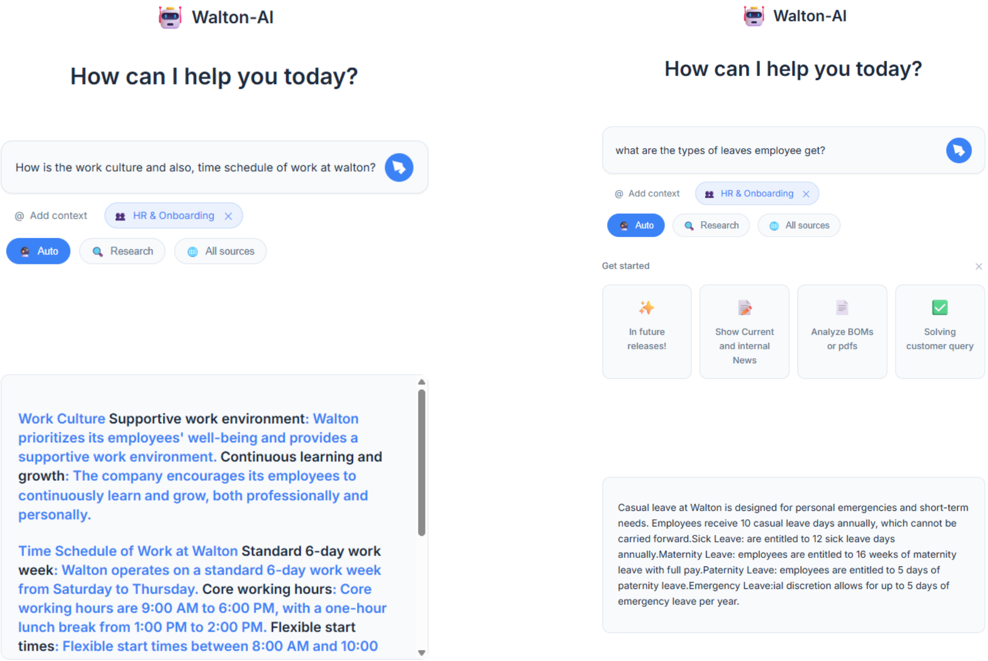
Query: "what are the types of leaves employee get?"
Analysis: Can generate long contexual answer.

5.2.3 Use Case 3: Sales Information

Query: "what is the marketing strategy of model 236L?"

5.3 Immediate Benefits (Present)

This project isn't just a technical exercise; it delivers immediate, measurable value to Walton's daily operations.



How is the work culture and also, time schedule of work at walton?

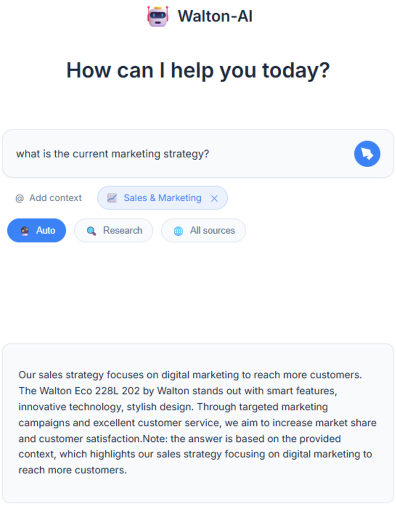
Work Culture Supportive work environment: Walton prioritizes its employees' well-being and provides a supportive work environment. Continuous learning and growth: The company encourages its employees to continuously learn and grow, both professionally and personally.

Time Schedule of Work at Walton Standard 6-day work week: Walton operates on a standard 6-day work week from Saturday to Thursday. Core working hours: Core working hours are 9:00 AM to 6:00 PM, with a one-hour lunch break from 1:00 PM to 2:00 PM. Flexible start times: Flexible start times between 8:00 AM and 10:00

what are the types of leaves employee get?

Casual leave at Walton is designed for personal emergencies and short-term needs. Employees receive 10 casual leave days annually, which cannot be carried forward.Sick Leave: are entitled to 12 sick leave days annually.Maternity Leave: employees are entitled to 16 weeks of maternity leave with full pay.Paternity Leave: employees are entitled to 5 days of paternity leave.Emergency Leave:ial discretion allows for up to 5 days of emergency leave per year.

Figure 5.3: Use Case 2: HR Policy Inquiry



what is the current marketing strategy?

Our sales strategy focuses on digital marketing to reach more customers. The Walton Eco 228L 202 by Walton stands out with smart features, innovative technology, stylish design. Through targeted marketing campaigns and excellent customer service, we aim to increase market share and customer satisfaction.Note: the answer is based on the provided context, which highlights our sales strategy focusing on digital marketing to reach more customers.

Figure 5.4: Use Case 3: Sales and Marketing

5.3.1 Productivity Gains

By providing instant, specific knowledge, Walton-AI dramatically cuts down on wasted time. A customer service agent or field technician can now get a precise answer to a technical question in seconds, rather than the 15-20 minutes they might have spent searching through manuals or waiting for a senior colleague to become available. Across hundreds of employees, this translates to thousands of saved work-hours per month, allowing our teams to resolve more customer issues, faster.

5.3.2 Security Enhancement

The most critical immediate benefit is closing a major security loophole. Before Walton-AI, there was a significant risk of employees pasting sensitive information—customer details, internal schematics, or unique repair solutions—into public AI websites. Our system eliminates this risk entirely. Because it runs 100% locally, Walton's proprietary data and intellectual property are kept secure and confidential, where they belong.

5.3.3 Cost Savings

Faster problem resolution directly translates to cost savings. Technicians can complete service calls more efficiently, reducing labor costs and potentially avoiding the need for follow-up visits. Furthermore, Walton-AI acts as an expert training assistant. New hires can learn the ropes by asking the system questions, significantly reducing the time senior staff must spend on onboarding and mentoring.

5.4 Long-Term Strategic Value (Future)

While the immediate benefits are compelling, the true power of Walton-AI lies in its potential as a strategic platform for future growth and innovation.

5.4.1 Scalable Knowledge Asset

Today, the system is an expert on refrigerators. Tomorrow, it will be an expert on air conditioners, televisions, and smartphones. The architecture is designed to grow. As we feed it more documents from every department—from R&D and manufacturing to HR and sales—Walton-AI will evolve into the company's central brain. It will become the single, reliable source of truth for the entire organization, preserving decades of institutional knowledge in a living, accessible format.

5.4.2 Foundation for Future AI Initiatives

This secure, in-house platform is the launchpad for a new generation of AI tools at Walton. The possibilities are extensive:

- **Customer-Facing Bots:** We can deploy a version on our website to provide customers with 24/7 support for common issues.

- **Personalized AI Assistants:** Using the long-term memory architecture, Walton-AI can remember interactions with specific employees, learning their roles and anticipating their needs.

- **Internal Process Automation:** The AI can be tasked with summarizing weekly sales reports, analyzing service data to spot recurring product faults, or even drafting internal communications.

- **Predictive Maintenance:** By analyzing years of complaint data, the system could eventually predict potential issues in new product lines before they become widespread problems.

5.4.3 Sustainable Competitive Advantage

In the modern market, the company that learns and adapts the fastest wins. Walton-AI gives us a powerful engine for learning. It makes our entire organization more efficient, more secure, and more intelligent. This is not a tool our competitors can simply buy; it is a unique, proprietary asset built on our own data and expertise. This strategic investment in our internal intelligence will be a key driver of our competitive advantage in the Bangladeshi market and beyond for years to come.

Chapter 6

Conclusion

This project began with a critical challenge: Walton's vast internal knowledge was locked away in scattered documents, making it difficult to access and creating a significant security risk in the age of public AI. In response, we developed Walton-AI, a secure, private, and intelligent assistant. By leveraging a state-of-the-art RAG architecture, the system delivers accurate, instant answers sourced directly from our own data, effectively transforming our company's knowledge from a scattered liability into a powerful, centralized asset.

We are proud to report that this project has successfully met every objective we set. We achieved remarkable accuracy in our tests, built a system that is fundamentally secure by design, delivered answers with impressive speed, and wrapped it all in an interface that is simple for anyone to use. Walton-AI is proof that we can embrace cutting-edge AI innovation while staying true to our core business needs of security, efficiency, and reliability.

Ultimately, Walton-AI is more than just a successful project; it is the first step toward a more intelligent and efficient future for our company. It represents a foundational investment in our own data and our own people. This system will not only improve how we work today but will serve as the cornerstone for the next generation of AI-powered tools, cementing Walton's position as an innovative leader in its industry for many years to come.

Bibliography

[1] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, H. Wang, and H. Wang, "Retrieval-augmented generation for large language models: A survey," *arXiv preprint arXiv:2312.10997*, vol. 2, no. 1, 2023.

[2] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, and H. Wang, "Retrieval-augmented generation for large language models: A survey," 2024.

[3] S. Transformers, "all-MiniLM-L6-v2: Sentence embedding model (hugging face)." https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2, 2024. Hugging Face model card, License: Apache-2.0; accessed September 10, 2025.

[4] M. AI, "Llama-3.2-3B-Instruct (hugging face model)." https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct, 2024. Hugging Face model card, accessed September 10, 2025; Released September 25, 2024.

[5] G. Gerganov and contributors, "llama.cpp: Llm inference in c/c++." https://github.com/ggml-org/llama.cpp, 2025. GitHub repository, accessed September 10, 2025.