A HYBRID INTENT-DRIVEN AND RETRIEVAL-AUGMENTED CONVERSATIONAL FRAMEWORK FOR RELIABLE UNIVERSITY INFORMATION ACCESS

NICOLE MARY JOHNSON

S23B23/020

HUMPHERY MUBIRU

S23B23/035

ETHAN ARIKO

S23B23/007

A DISSERTATION SUBMITTED TO THE FACULTY OF ENGINEERING, DESIGN AND TECHNOLOGY IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF A DEGREE OF BACHELOR OF SCIENCE IN COMPUTER SCIENCE OF UGANDA CHRISTIAN UNIVERSITY

May, 2026

UGANDA CHRISTIAN UNIVERSITY

A Centre of Excellence in the Heart of Africa

Abstract

Students at Uganda Christian University frequently struggle to access timely and accurate institutional information because relevant details are scattered across notice boards, social media groups, and outdated web pages. This report presents the design, implementation, and evaluation of a hybrid multi-channel chatbot that merges intent-based routing, keyword retrieval, vector-based semantic search, and large-language-model generation to deliver context-grounded answers drawn from seventeen verified university documents. The system was deployed as both a web application and a WhatsApp integration, offering multilingual support in seven languages through the Sunbird translation service. Functional testing across representative query categories showed that the hybrid retrieval strategy returned relevant context for the majority of test queries, while a confidence-aware fallback mechanism directed users to appropriate university offices when the knowledge base could not provide a reliable answer. The findings indicate that combining complementary retrieval methods with grounded generation is a viable approach for building dependable information assistants in resource-constrained university settings.

Declaration

We, Nicole Mary Johnson, Mubiru Humphery, and Ariko Ethan, hereby declare that this is our original work, is not plagiarised and has not been submitted to any other institution for any award.

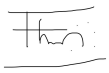
Nicole Mary Johnson

Signature:  Date: 28/05/2026

Mubiru Humphery

Signature:  Date: 28/05/2026

Ariko Ethan

Signature:  Date: 28/05/2026

Ethical Generative AI Usage Declaration

We, Nicole Mary Johnson, Mubiru Humphery, and Ariko Ethan, acknowledge that we used a generative AI assistant as a study buddy during this project in the manner described in Appendix A. The appendix discloses the AI tool used, the purposes for which it was used, and the major prompts issued. No content generated by AI was used verbatim in this final submission; all outputs were modified, verified, and integrated by our team.

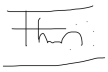
Nicole Mary Johnson

Signature:  Date: 28/05/2026

Mubiru Humphery

Signature:  Date: 28/05/2026

Ariko Ethan

Signature:  Date: 28/05/2026

Approval

This Research Report has been submitted for examination with the approval of the following supervisors.

Signature: _____ Date: 01/06/2026

Dr. Samuel Kakuba

Signature: _____ Date: 02 / 06 / 2026

Eng. Ian Raymond Osolo

Faculty of Engineering, Design and Technology
Department of Computing and Technology
Uganda Christian University

Dedication

We dedicate this work to our families, whose encouragement sustained us throughout our studies, and to the students of Uganda Christian University, for whom this system was built.

Acknowledgements

We are grateful to our supervisor and the Department of Computing and Technology at Uganda Christian University for their guidance during this project. We also acknowledge the open-source communities behind the software libraries used in this work, as well as organizations like Sunbird AI, and Meta for providing the cloud services and APIs that made the deployment feasible.

Contents

List of Tables

List of Figures

List of Acronyms and Abbreviations

API	Application Programming Interface
BM25	Best Matching 25 (probabilistic keyword ranking)
GDPR	General Data Protection Regulation
LLM	Large Language Model
NLP	Natural Language Processing
OCR	Optical Character Recognition
ONNX	Open Neural Network Exchange
RAG	Retrieval-Augmented Generation
REST	Representational State Transfer
SQL	Structured Query Language
SSE	Server-Sent Events
STT	Speech-to-Text
TTS	Text-to-Speech
UCU	Uganda Christian University
WSGI	Web Server Gateway Interface

Chapter One

Introduction

1.1 Background to the Study

Universities worldwide adopt digital tools to improve the speed and consistency of administrative communication with students. At many institutions, however, essential information such as fee structures, registration deadlines, grading policies, and campus directions remains scattered across physical notice boards, disparate web pages, and informal messaging groups. This fragmentation forces students to invest considerable time and effort searching for basic factual answers, while administrative staff repeatedly address the same set of questions each academic term. Within sub-Saharan Africa, these challenges are amplified by intermittent internet connectivity, multilingual student populations, limited computing infrastructure, and the near-total absence of purpose-built digital assistants tailored to local institutional data. Recent advances in retrieval-augmented generation (RAG) have demonstrated that coupling a document retrieval step with a large-language-model generation step can produce accurate, source-grounded responses to factual queries. Yet most documented RAG deployments target well-resourced Western universities and do not account for the constraints encountered in an African higher-education setting.

Uganda Christian University (UCU) is a private chartered institution founded in 1997 on the heritage of Bishop Tucker Theological College, which dates back to 1913. The university serves a diverse student body across undergraduate and postgraduate programmes in multiple faculties. Like many growing African universities, UCU has accumulated institutional information in a variety of formats and platforms over the years: printed handbooks, PDF policy documents, an academic portal, and faculty-specific social media groups. Students, particularly those newly admitted, routinely report difficulty in locating accurate answers to common questions about fees, deadlines, and campus facilities. Continuing students face similar challenges when regulations change between academic years. Administrative offices, in turn, devote substantial portions of their working hours to answering repetitive inquiries, which diverts resources from more complex support tasks. The emergence of cloud-based language models and lightweight embedding techniques has made it possible to build information assistants that operate within the computational and financial constraints of a small univer-

sity. Sentence-transformer models such as MiniLM-v2 [1] provide near-state-of-the-art embedding quality at a fraction of the cost of larger models, and hosted inference services such as Groq offer low-latency generation without requiring on-premises GPU hardware. Meanwhile, open translation initiatives such as Sunbird AI supply language-identification and translation endpoints for several Ugandan languages, opening a pathway for genuinely multilingual student support. This project addressed the gap by building and evaluating a hybrid multi-channel chatbot for Uganda Christian University.

1.2 Problem Statement

UCU presently lacks a single, reliable channel through which students can obtain factual answers to routine university-related questions; information is fragmented across the Alpha academic portal, physical notice boards, WhatsApp groups, and occasionally outdated web pages, leading to inconsistent answers, delays during peak registration periods, misinformation propagated through informal channels, a disproportionate administrative burden on support staff, and the consistent failure of commercially available chatbot platforms and general-purpose language models on authentic UCU queries that contain local abbreviations, spelling variations, student slang, and code-switching between English and Ugandan languages, with no existing system combining grounded document retrieval, multilingual African-language support, multi-channel delivery, and real institutional knowledge in a single deployable framework.

1.3 Objectives of the Study

1.3.1 Main Objective

To design, implement, and evaluate a hybrid multi-channel university chatbot that provides reliable, timely, and context-grounded answers from institutional knowledge sources.

1.3.2 Specific Objectives

1. To design and implement an end-to-end hybrid chatbot that integrates intent routing, lexical retrieval, vector retrieval, and large-language-model generation over verified UCU institutional documents.

2. To build and deploy a multi-channel interaction workflow covering web and WhatsApp, and to verify cross-channel functional parity on a common set of test queries.

3. To define and apply an evaluation framework that measures retrieval hit rate, generation faithfulness, fallback-routing classification performance, end-to-end latency, and round-trip translation quality on the deployed system.

1.4 Research Questions

The research questions below are aligned with the evaluation actually carried out in Chapter 4. An earlier version of this study included a question on the chatbot's effect on staff workload; that question was removed because workload reduction could not be measured within the scope of the project, and a question on retrieval behaviour across query categories (which the evaluation does answer) was put in its place.

1. How effectively can a hybrid pipeline that combines intent routing, lexical retrieval, vector retrieval, and large-language-model generation deliver context-grounded answers across the major UCU information categories?

2. How do retrieval coverage, generation faithfulness, and the confidence-aware fallback behave across the principal query types handled by the deployed system (in-scope text, out-of-scope, multilingual, and vision)?

3. How suitable is the architecture for multi-channel deployment, measured in terms of cross-channel functional parity and end-to-end response latency?

1.5 Rationale of the Study

The motivation for this work arose from the day-to-day observation that students at UCU spend a disproportionate amount of time seeking basic factual information, while administrative staff repeatedly answer the same procedural questions. An automated assistant grounded in verified documents could in principle shorten the time-to-answer for routine queries from hours or days to seconds. Because the majority of UCU students communicate through WhatsApp on a daily basis, integrating such an assistant into that channel could maximise adoption without requiring any additional software installation. The rationale of the study is therefore the practical motivation: addressing an information-access bottleneck that students and staff encounter directly.

1.6 Justification of the Study

While the rationale above explains why the project was undertaken from a local-needs perspective, the justification explains why it is a defensible contribution to the academic

literature on educational chatbots. Three gaps in the current literature support the choice of artefact.

First, the surveyed deployments of retrieval-augmented generation in educational settings are concentrated in well-resourced Western universities, and the few published systems oriented to policy or institutional documents, most notably Nagarajan, Kumar, and Santhiappan [2], have not been evaluated in African higher-education contexts. Okonkwo and Ade-Ibijola [3] and Peyton, Unnikrishnan, and Mulligan [4] both explicitly call for educational chatbot studies that move beyond laboratory prototypes and report on real-world deployments in under-resourced settings. The present work is one such deployment.

Second, the hybrid systems documented in the educational-chatbot literature, for example Mangotra, Dabas, Khetharpal, *et al.* [5], Kathole, Patil, Jadhav, *et al.* [6], and Zaidi, Ahmed, Husain, *et al.* [7], are built around intent classification as the primary answering mechanism, which constrains them to the intents they were trained on and provides no graceful behaviour outside that set. Substituting a retrieval-and-generation pipeline for the classifier is a meaningful architectural shift that the literature has identified as desirable [8], [9] but has rarely instantiated in a deployed university chatbot.

Third, no system surveyed in the review combines grounded RAG retrieval with African-language translation pivoting and mobile-messaging delivery in a single artefact. Lai and Lee [10] report that the vast majority of conversational AI systems in education operate in English alone, and large-scale work on low-resource translation [11], [12] has not yet been carried through into deployed student-facing services in sub-Saharan Africa. Building such a system therefore makes a verifiable contribution at the intersection of three currently disjoint literatures.

1.7 Significance of the Study

The significance of the study, distinct from its rationale and academic justification, lies in three concrete artefacts that other institutions can reuse. First, the codebase provides a working reference implementation of a confidence-gated, grounded-generation pipeline that can be re-pointed at any other institutional document set with no model retraining. Second, the office-routing fallback design pattern, which is invoked when no relevant context is retrieved, provides a deployable mechanism for trading coverage against reliability, which is critical when incorrect information about fees or regulations carries real consequences. Third, the multilingual layer demonstrates that an English-only knowledge base can serve an African-language student population through translation pivoting, lowering the data-requirement barrier for similar institutions whose policy documents exist only in English.

1.8 Scope of the Study

The study was conducted at Uganda Christian University, Mukono campus. UCU was selected because it presented all the information-access challenges described in the literature for small to medium-sized African universities: fragmented digital resources, multilingual student intake, and constrained ICT budgets. The knowledge base comprised seventeen institutional documents covering admissions, fees, academic calendars, grading systems, student regulations, campus buildings, and related policies. The deployment channels were a responsive web application and the WhatsApp messaging platform via the Meta Cloud API. Multilingual support was provided for English, Luganda, Acholi, Ateso, Lugbara, Runyankole, and Swahili. The evaluation focused on functional correctness, retrieval relevance, response latency, and fallback behaviour; a large-scale user satisfaction survey was beyond the scope of this phase of the project.

1.9 Conceptual Framework

The conceptual framework for this project rests on three interacting layers. The first layer is the knowledge layer, which encompasses the ingestion, preprocessing, chunking, and indexing of institutional documents into both a vector database and a lexical search index. The second layer is the retrieval and generation layer, which receives a user query, performs hybrid retrieval (combining semantic similarity search with keyword matching), assembles the most relevant context chunks, and passes them to a large language model that generates a grounded, natural-language answer. The third layer is the delivery and monitoring layer, which manages multi-channel dispatch (web and WhatsApp), multilingual translation, image-based analysis, and an analytics dashboard that records interaction quality metrics.

The key theoretical underpinning is the retrieval-augmented generation paradigm introduced by Lewis, Perez, Piktus, *et al.* [8], which argues that coupling a non-parametric retrieval component with a parametric generator yields more faithful and verifiable responses than either component alone. The project extends this paradigm by adding a lexical retrieval branch, a keyword-based intent routing step, and a confidence-gated fallback mechanism, forming what we term a *hybrid* conversational framework.

1.10 Report Summary

Chapter 1 has presented the background, problem statement, objectives, research questions, rationale, justification, significance, scope, and conceptual framework. Chapter 2 reviews the theoretical and empirical literature on educational chatbots and retrieval-

augmented generation. Chapter 3 describes the research methodology, including the design science approach, data preparation procedures, and system architecture. Chapter 4 presents the implementation details and the results of functional testing. Chapter 5 discusses the findings in relation to the literature and the research questions. Chapter 6 draws conclusions, states limitations, and proposes directions for future work.

Chapter Two

Literature Review

2.1 Introduction

This chapter examines the body of work that informed the design and evaluation of the chatbot system. The review is organised into three strands: a theoretical review of the key techniques and paradigms, an empirical review of existing educational chatbot implementations, and an identification of the research gap that the present study addresses.

2.2 Theoretical Literature Review

2.2.1 Evolution of Educational Chatbots

Early university chatbot deployments relied on rule-based or pattern-matching architectures. Georgia State University's *Pounce* system, one of the most cited examples, used predefined decision trees to guide admitted students through pre-enrolment tasks and achieved measurable reductions in "summer melt" [3]. Rule-based systems offered predictability but scaled poorly: every new question type demanded manual authoring of additional rules.

The introduction of transformer-based language models after 2017 shifted the landscape. Wollny, Schneider, Di Mitri, *et al.* [13] conducted a systematic literature review of chatbots in education and found that, by 2021, an increasing share of new systems leveraged neural models for intent classification or response generation. The post-2020 period saw an explosive growth driven by large language models hosted in the cloud, coinciding with pandemic-era demand for online student services [14].

2.2.2 Retrieval-Augmented Generation

Lewis, Perez, Piktus, *et al.* [8] formalised retrieval-augmented generation as a framework in which a retrieval module first selects relevant passages from a knowledge store and a sequence-to-sequence generator then conditions its output on those passages. The approach addressed a core weakness of purely generative models: their tendency to hallucinate plausible but factually incorrect details. Ovadia, Brief, Mishaeli, *et al.* [9] later compared RAG with fine-tuning for knowledge injection and concluded that RAG was

superior when the underlying knowledge base changed frequently, which is precisely the situation at a university where policies and fee structures are updated each academic year.

Nagarajan, Kumar, and Santhiappan [2] demonstrated the applicability of RAG to institutional policy documents through their *PolicyBot* system, which answered questions over organisational policy corpora with high factual accuracy. Their work confirmed that grounding generation in retrieved evidence significantly reduces the rate of unsupported assertions.

2.2.3 Lightweight Embedding Models

Deploying embedding-based retrieval at scale requires models that balance quality against computational cost. Wang, Bao, Huang, *et al.* [1] proposed MiniLM-v2, a distillation technique that transfers multi-head self-attention relations from a large teacher model to a compact student model. The resulting `all-MiniLM-L6-v2` model achieves competitive performance on standard sentence-similarity benchmarks while requiring a fraction of the memory and inference time of full-sized BERT variants. This characteristic makes it well suited to free-tier cloud deployments where RAM and CPU are tightly constrained.

2.2.4 Vector Databases and Hybrid Retrieval

Rusum and Anasuri [15] surveyed the role of vector databases in modern applications and highlighted their growing importance for real-time semantic search and RAG pipelines. They noted that combining vector-based retrieval with traditional keyword-based methods can mitigate the weaknesses of either approach in isolation: vector search excels at capturing paraphrases and semantic equivalences, while keyword search is effective when the query contains specific codes, names, or numbers that carry precise meaning.

2.2.5 Multilingual and Accessibility Considerations

Lai and Lee [10] conducted a systematic review of conversational AI tools in education and found that the vast majority of documented systems operated exclusively in English. The handful of multilingual systems relied on machine translation as a bridge, a strategy that introduces translation error but remains the most practical option when training data in the target language is scarce.

The bridge-translation approach has been reshaped by recent large-scale low-resource translation work. The No Language Left Behind project [11] released a single 54B-parameter encoder–decoder model covering more than two hundred languages, includ-

ing several Bantu and Nilotic languages relevant to East African universities, and reported substantial improvements over prior open systems on low-resource pairs. The model still under-performs on languages with very limited parallel data, but it establishes pivoting through a strong multilingual model as a credible default for systems that cannot afford language-specific fine-tuning.

Parallel to this, the Masakhane initiative has produced Africa-centric resources and benchmarks for natural-language processing. Adelani, Neubig, Ruder, *et al.* [12] introduced *MasakhaNER 2.0*, a named-entity recognition benchmark across twenty African languages, and demonstrated that transfer learning from typologically related languages can substantially improve downstream task performance on under-represented languages. Their work argues that progress on African-language NLP requires both data curation and modelling techniques attuned to the typological properties of these languages, and provides empirical evidence that off-the-shelf multilingual models can be adapted with relatively modest parallel-data investments.

On the accessibility side, Horvat, Horvat, Havaš, *et al.* [16] explored the accessibility dimension of educational chatbots and argued that voice-based and mobile-friendly interfaces substantially broaden the user base, particularly for students with visual impairments or limited access to desktop computers. Reitmaier, Wallington, Kalarikalayil Raju, *et al.* [17] extended this argument empirically through a field study of automatic speech recognition for low-resource-language speakers, reporting that voice-first interfaces lower the literacy and typing barriers that often gate text-based digital services in multilingual sub-Saharan contexts, while also identifying outstanding challenges in code-switching and dialectal variation that voice systems must address to be useful in practice. Taken together, these strands of work motivate the present project's combined translation-pivot plus optional speech-input design: machine translation supplies coverage breadth, while voice input lowers the literacy threshold for students who prefer spoken interaction.

2.3 Empirical Literature Review

2.3.1 University Chatbot Deployments

Okonkwo and Ade-Ibijola [3] reviewed chatbot applications in education and identified administrative advising, FAQ answering, and student orientation as the dominant use cases. They observed that most systems were evaluated with small convenience samples and called for more rigorous and larger-scale evaluations.

Peyton, Unnikrishnan, and Mulligan [4] conducted a focused review of university chatbots for student support and found that the field was moving beyond simple FAQ retrieval toward more sophisticated architectures that could handle follow-up questions

and contextual references. They also noted that real-world deployment remained rare: most papers described prototypes tested only in laboratory conditions.

Ramakrishnan, Thangamuthu, Nguyen, *et al.* [18] described an NLP-powered university chatbot and reported that accuracy dropped sharply when queries contained informal, non-standard English. This finding underscored the importance of preprocessing and retrieval strategies that can cope with noisy input.

2.3.2 Intent Classification and Hybrid Approaches

Mangotra, Dabas, Khetharpal, *et al.* [5] built a university FAQ chatbot using a neural-network-based intent classifier and reported satisfactory accuracy on a curated set of intents. Kathole, Patil, Jadhav, *et al.* [6] proposed an adaptive intent-based system with a deep temporal convolutional network and demonstrated improved handling of ambiguous queries. Song, Lv, Zhu, *et al.* [19] described an intelligent campus consultation assistant that combined rule-based routing with neural generation and found that the hybrid approach outperformed either method alone on a campus-specific query benchmark.

Zaidi, Ahmed, Husain, *et al.* [7] developed a hybrid chatbot that paired SBERT encodings with a nearest-neighbour classifier for intent detection, achieving high intent-classification accuracy on a university query dataset. Their work confirmed that sentence-level embeddings could serve as effective features for routing student queries to the correct processing pathway.

2.3.3 Chatbot Usability in Higher Education

Bringula, Jose, Lardizabal, *et al.* [20] investigated the predictors of usability for a mobile intelligent agent that provided information to college students. They found that perceived usefulness, ease of use, and information quality significantly predicted student satisfaction, suggesting that response accuracy is a necessary but not sufficient condition for adoption. These usability insights informed the interface design decisions in the present project, particularly the emphasis on source attribution and confidence-based fallback messages.

2.4 Research Gap

Each of the systems reviewed above addresses one or two of the dimensions required for a deployable UCU chatbot, but none addresses all of them. Reading the reviewed work as a list of partial solutions makes the gap more concrete than a general statement can.

Georgia State University's *Pounce* system, as reviewed by Okonkwo and Ade-Ibijola [3], used a hand-authored rule base and a decision-tree flow that required manual addition of every new question type; it had no document grounding, no multilingual coverage, and was delivered through a single channel. Mangotra, Dabas, Khetharpal, *et al.* [5] replaced the rules with a BiLSTM and feed-forward neural intent classifier and added a confidence threshold of 60 that returns a default answer when the model falls below it, but the classifier was trained on a 75-query self-curated dataset and provided no mechanism for retrieving evidence from institutional documents. Kathole, Patil, Jadhav, *et al.* [6] improved intent disambiguation further through an adaptive deep temporal convolutional network optimised by the ADT-BMO algorithm, but still answered from a fixed response set rather than from the live document corpus, and operated only in English. Song, Lv, Zhu, *et al.* [19] described a Seq2Seq deep-learning consultation assistant trained on a Chinese campus corpus, delivered through a single Web Chat front end, with neither a lexical retrieval branch nor retrieval-augmented generation over institutional documents. Zaidi, Ahmed, Husain, *et al.* [7] combined Semantic BERT (SBERT) with TF-IDF and cosine similarity over a 500-query CSV of student questions, which is genuinely a semantic-plus-lexical hybrid in the retrieval sense, but the retrieval target was a curated question-answer set rather than institutional documents and the system was deployed through a single Streamlit front end with no out-of-scope fallback.

The closest baseline in the surveyed literature is Nagarajan, Kumar, and Santhiappan [2], whose *PolicyBot* system applies multi-stage dense retrieval (HyDE, multi-query expansion, Reciprocal Rank Fusion, and cross-encoder reranking) to organisational policy documents and includes an explicit "not enough context" fallback when the retrieved evidence is insufficient. The retrieval pipeline is entirely dense rather than hybrid lexical-and-vector, the deployment uses an English policy corpus through a single Streamlit interface, and the fallback message is a single generic refusal rather than a routing decision that names a specific office contact. Ramakrishnan, Thangamuthu, Nguyen, *et al.* [18] explicitly observed that NLP-powered university chatbots degrade sharply on noisy or code-switched input, a failure mode that none of the surveyed systems mitigates through a hybrid lexical-and-semantic retrieval design.

The present study is the first system surveyed in this review that simultaneously (a) grounds answers through hybrid lexical-and-vector retrieval over institutional documents rather than through intent classification, (b) supports African-language students through Sunbird translation pivoting rather than English-only operation, (c) is delivered through both a web interface and a widely used mobile-messaging channel rather than a single front end, and (d) routes unanswerable queries through a confidence-gated fallback to the appropriate named UCU offices rather than fabricating an answer or returning a generic apology. The remainder of this report describes that artefact and evaluates it against the framework defined in Chapter 3.

Chapter Three

Research Methodology

3.1 Introduction

This chapter describes the methodological approach adopted for the project, the data collection and preparation procedures, the system architecture, and the evaluation strategy. Each design decision is explained with reference to the constraints and objectives stated in Chapter 1.

3.2 Research Design

The project followed a Design Science Research approach, which is appropriate when the goal is to create and rigorously evaluate a technological artefact that addresses an identified organisational problem. Design science proceeds through iterative cycles of problem identification, artefact design, implementation, and evaluation, with each cycle refining the artefact based on observed shortcomings. Six phases were carried out:

1. **Problem and requirements analysis.** Information-access bottlenecks at UCU were identified through observation and informal consultation with students and administrative staff. Core requirements were defined: accuracy, speed, reliability, and multi-channel support.

2. **Data preparation.** Institutional documents were collected, digitised where necessary, cleaned, and chunked into retrieval units.

3. **Knowledge layer construction.** Vector embeddings were generated with a sentence-transformer model and indexed in a cloud-hosted vector database; a parallel lexical index was built for keyword-based retrieval.

4. **Hybrid answer pipeline development.** An orchestration layer was implemented that combined intent routing, lexical and vector retrieval, context assembly, large-language-model generation, and confidence-based fallback.

5. **Integration and deployment.** The pipeline was connected to a Flask-based web interface and to WhatsApp via the Meta Cloud API, then deployed on a cloud platform.

6. **Evaluation and refinement.** The system was tested for retrieval relevance, response correctness, latency, and fallback behaviour; deficiencies were corrected through further iterations. The refinement loop also incorporated an informal user-feedback round with between fifteen and twenty UCU students who used the deployed system; the substance of that round, and the feature refinements it drove, are reported in Section 4.6.

3.3 Sources of Information

3.3.1 Primary Sources

The primary data source was a corpus of seventeen official UCU institutional documents. These documents covered admissions procedures, fee structures and deadlines, the academic calendar, grading systems, graduation requirements, the student code of conduct, postgraduate and undergraduate regulations, campus building information, the university's research policy, and a curated knowledge base of frequently asked questions. The documents were obtained from official university publications and verified against current academic-year information.

Table 3.1 lists the documents that constituted the knowledge base.

3.3.2 Secondary Sources

Secondary sources included the academic literature reviewed in Chapter 2 and the technical documentation for the open-source libraries and cloud services used in the implementation.

3.4 Data Collection Instruments

The instruments for data collection consisted of:

- A document ingestion pipeline capable of loading PDF and plain-text files, applying optical character recognition where needed, and extracting clean text.
- A logging module embedded in the web application that recorded every user interaction, including the query text, the generated answer, the source language, whether translation was applied, whether the fallback mechanism was triggered, and the end-to-end response latency.

Table 3.1: Institutional documents in the knowledge base. The seventeen documents listed below were the entire primary corpus over which both the lexical index and the Qdrant vector index were built. The selection was deliberately limited to publicly published UCU material covering admissions, fees and deadlines, the academic calendar, grading and graduation regulations, the student code of conduct, undergraduate and postgraduate regulations, campus building information, the university's 2020 Policy on Research, UCU's administrative directory, the curated UCU Knowledge Base (FAQ), the University Information Systems guide, the Moodle LMS guide, the Computing Sciences and Engineering Students' Association (CSEA) handbook, and the institutional timeline. These categories define the in-scope coverage that Chapter 4's evaluation is measured against; any student query falling outside them is expected to trigger the fallback mechanism described in Section 3.6.

No.	Document Title
1	Admission Information
2	Advent 2026 Fees and Deadlines
3	Fee Structure 2025 Intake
4	Dates for UCU Academic Calendar 2026
5	Grading System
6	Graduation Requirements
7	Student Code of Conduct
8	Undergraduate Student Handbook
9	Postgraduate Regulations
10	2020 Policy on Research
11	UCU Administration
12	UCU Buildings
13	UCU Knowledge Base (FAQ)
14	University Information Systems
15	Moodle LMS Guide
16	Computing Sciences and Engineering Students' Association (CSEA)
17	Timeline Since Establishment

3.5 Data Processing and Analysis

3.5.1 Document Preprocessing

Each document was loaded through a file-type-specific loader. PDF files were parsed page by page using the `pypdf` library; when a page yielded little or no extractable text, an OCR fallback converted the page image to text using Tesseract at 300 DPI in greyscale mode. After extraction, the full text of every document was saved to a plain-text cache to support the lexical search component.

3.5.2 Chunking Strategy

The cleaned text was split into overlapping chunks using a word-level sliding window. Each chunk contained up to 300 words, and consecutive chunks overlapped by 60 words (a 20% overlap). This overlap ensured that contextual information near chunk boundaries was not lost. The process yielded 284 chunks across the seventeen documents.

Justification of chunk parameters. The 300-word setting was chosen to keep a single complete policy paragraph, fee-structure block, or academic-calendar entry inside one retrieval unit, so that the language model receives a coherent reasoning context rather than fragmented sentences. The value sits within the 100–500-word range that is conventional in the dense-retrieval literature for similar narrative documents [8], [21]. The 60-word (20%) overlap was empirically selected because clauses in UCU regulations frequently span paragraph boundaries. For instance, a fee figure is stated in one paragraph and qualified in the next, and a code-of-conduct rule is introduced in one section and exemplified in the following section. A non-overlapping segmentation produced chunks that lost the antecedent of the qualifying clause during pilot retrieval tests. The `all-MiniLM-L6-v2` encoder used in the embedding step operates on an effective 256-token context window, which is shorter than the 300-word chunk size; chunks that exceed this length are therefore truncated by the encoder. This truncation is mitigated by the parallel lexical search, which scans the full text cache and recovers any keyword matches that fall in the truncated tail of a chunk. The hybrid retrieval design therefore tolerates the slightly oversized chunks without losing recall, which would not be true of a vector-only system.

3.5.3 Embedding and Indexing

Each chunk was encoded into a dense vector representation using the `all-MiniLM-L6-v2` sentence-transformer model, executed through the FastEmbed library with the ONNX runtime to minimise inference cost. The resulting vectors were uploaded to a Qdrant cloud instance and stored in a collection named `ucu_docs` with cosine similarity as the distance metric.

3.5.4 Lexical Index

In parallel, a keyword-based index was constructed over the plain-text cache files. At query time, the lexical search module removed standard English stopwords from the query, scanned every line in the cache for matching terms, scored each hit by the number of distinct query terms found within a context window of five lines above and below, and returned the top eight results ranked by score.

3.6 System Architecture

The architecture comprised three layers, consistent with the conceptual framework described in Chapter 1.

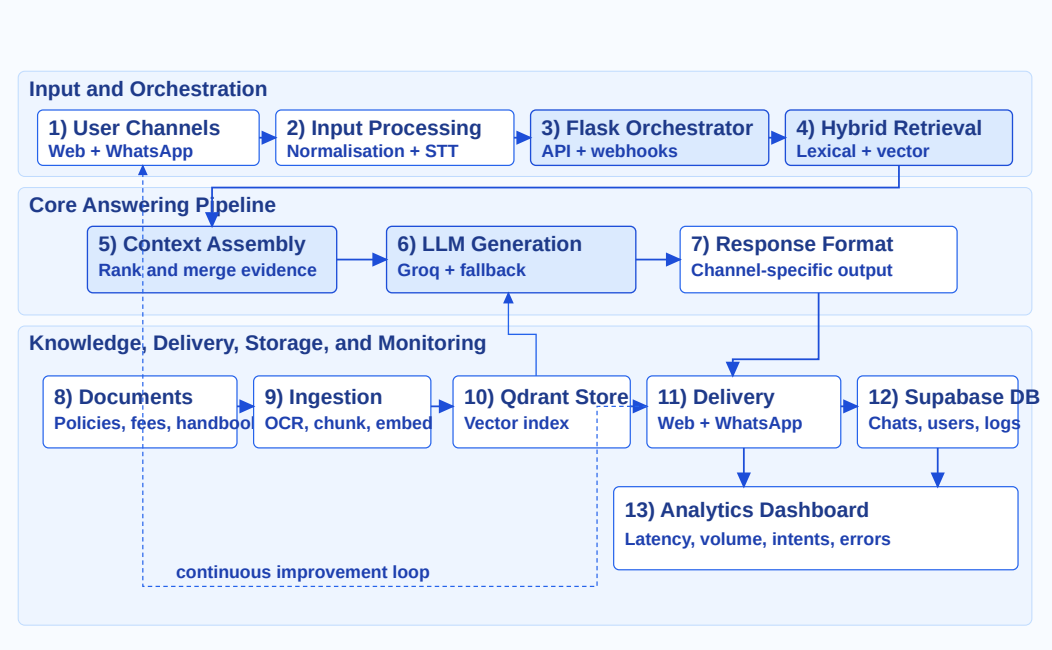


Figure 3.1: End-to-end system architecture. The three layers shown, namely input and orchestration (top), the core answering pipeline (middle, comprising hybrid retrieval, context assembly, generation, and channel-specific formatting), and knowledge, delivery, storage, and monitoring (bottom), correspond to the three layers of the conceptual framework introduced in Section 1.9. Numbered components (1)–(13) trace the path of a single user query from arrival on a channel, through retrieval and generation, to response delivery and to the analytics dashboard. The dashed feedback line denotes the continuous-improvement loop in which interaction logs (component 13) inform refinements to the knowledge base (component 8) and to the orchestration prompts (component 6). The figure shows that no component is duplicated per channel, since the same retrieval and generation core serves both the web and WhatsApp front ends, which is the architectural property that supports the cross-channel functional-parity claim restated in Conclusion 2 of Chapter 6.

3.6.1 Input and Orchestration Layer

Users interact with the system through two channels: a browser-based web application and WhatsApp. The web application is built with Flask and provides a responsive chat interface, a sidebar displaying previous conversations, and an administrative analytics dashboard. The WhatsApp channel is connected through the Meta Cloud API; inbound messages are delivered to a webhook endpoint hosted on the same Flask application.

When a message arrives, the orchestration layer determines the input modality. Text

queries proceed to the retrieval and generation pipeline. Image queries are routed to a vision analysis module that classifies the image scenario (dress-code check, building identification, document reading, or general) based on keywords in the accompanying question, and then sends the image along with scenario-specific context to a multimodal language model.

Multilingual handling occurs at the boundaries: inbound non-English text is detected using the Sunbird language-identification endpoint and translated to English before retrieval; the generated English answer is translated back to the user's source language before delivery. Seven languages are supported: English, Luganda, Acholi, Ateso, Lugbara, Runyankole, and Swahili.

3.6.2 Core Answering Pipeline

The retrieval step runs two parallel searches:

1. **Semantic search.** The user query is embedded with the same `all-MiniLM-L6-v2` model and compared against the Qdrant vector index. The six nearest chunks by cosine distance are returned.

2. **Lexical search.** The query is tokenised, stopwords are removed, and the keyword index is scanned to return up to eight matching text windows ranked by term-hit density.

The results are merged by alternating one semantic result with one lexical result, de-duplicated by comparing the first 200 characters of each chunk, and truncated to four chunks. These four chunks are formatted with their source titles and passed to the Groq-hosted `llama-3.1-8b-instant` language model, which generates a natural-language answer. The generation prompt instructs the model to rely exclusively on the provided context, cite sources naturally within the text, include specific numbers and dates where present in the context, and refrain from fabricating information.

If the retrieval step returns no relevant chunks, the system invokes a fallback module that maps query keywords to one of seven UCU offices (Admissions, Registrar, Finance, Library, ICT Helpdesk, Health Centre, Student Affairs) and directs the user to the appropriate contact point.

3.6.3 Knowledge, Delivery, Storage, and Monitoring

Generated answers are dispatched through the originating channel. For the web interface, the response is returned as JSON and rendered in the chat pane. For WhatsApp, the response is sent via the Meta Graph API. If the query matched a known campus building, the system also sends a photograph of that building; this visual response draws on a curated gallery of 28 images covering sixteen campus locations.

Every interaction is logged to a PostgreSQL database (Supabase-hosted in production, SQLite for local development). The logged fields include the channel, user reference, query text, answer text, detected source language, translation flags, success or failure status, whether the fallback was used, and the response latency in milliseconds. An administrative dashboard aggregates these records into key performance indicators: total interactions, success rate, fallback rate, error rate, average answer length, median and 95th-percentile latency, multilingual query volume, and per-language translation counts. The dashboard also generates a heatmap of interaction density by day-of-week and hour-of-day.

3.7 Evaluation Strategy

This section describes the evaluation framework that governs Chapter 4. It is set out here, in the methodology, so that the results reported later in this report can be measured against pre-stated criteria rather than narrated post-hoc.

3.7.1 Test Query Construction and Sampling Rationale

A bench of fifty test queries was assembled to exercise the system across the dimensions on which it is expected to perform. The sampling strategy was *stratified* rather than random, because the knowledge base is partitioned into well-defined topical categories of unequal size and because the system's behaviour on each category is qualitatively different (paraphrase tolerance, exact-figure recall, fallback routing, and translation round-tripping). A random sample over all possible student questions would have under-represented the rare categories on which failure modes are most informative. The bench therefore comprised:

- **In-scope text (35 queries):** five queries per topical category, covering admissions, fees and deadlines, the academic calendar, grading and graduation, student conduct, campus buildings, and postgraduate regulations. Within each category, the five queries were composed deliberately as one verbatim phrasing taken from the source document, one paraphrase, one query containing a precise figure or date (to test lexical recall), one noisy or abbreviation-heavy phrasing, and one code-switched English–Luganda phrasing (to test the lexical-and-semantic complementarity).
- **Out-of-scope (5 queries):** questions outside the knowledge base such as the weather, the day's news, or unrelated personal topics, used to assess fallback routing.

- **Multilingual (5 queries):** two Luganda, one Acholi, one Runyankole, and one Swahili paraphrase of in-scope queries, used to assess translation and retrieval round-trip behaviour.
- **Vision (5 queries):** two dress-code compliance checks, two building-identification photos, and one document-reading photo, used to verify the vision module's scenario routing and grounding.

The queries were sourced from common student questions surfaced during the Phase-1 informal consultation described in Section 3.2. Grounding the bench in real student questions is important because earlier evaluations of educational chatbots have been criticised for relying on artificial query sets that do not reflect the noisy, code-switched language students actually use [18].

3.7.2 Evaluation Metrics

Four families of metrics were used.

Retrieval metrics measure whether the relevant evidence was surfaced. The principal metric is the *top-k chunk hit rate*: the proportion of test queries for which at least one of the four chunks passed to the language model contains the answer text or a paraphrase thereof. A query whose answer is split across multiple chunks counts as a hit if any contributing chunk is present.

Generation quality metrics measure whether the answer is faithful to the retrieved evidence. *Faithfulness* is the proportion of factual claims in the answer that can be attributed to a retrieved chunk; it is computed by manually decomposing each answer into atomic claims and checking each against the chunks supplied to the model. The construct is taken from the RAGAS framework [22]. *Source-citation rate* is the proportion of answers that name the originating document either by title or by a natural reference (e.g. "According to the Student Code of Conduct…"). *ROUGE-L F1* [23] is reported between the generated answer and the relevant source paragraph as a coarse surface-overlap signal, with the explicit caveat that low ROUGE values are expected and acceptable because the model is instructed to rephrase rather than copy.

System-behaviour metrics treat the in-scope/out-of-scope decision as a binary classification problem and report *precision*, *recall*, and *F1* of the fallback routing, with "correctly routed to fallback" as the positive class for out-of-scope queries. End-to-end latency is reported as median and 95th percentile in milliseconds, read from the `interaction_events.latency_ms` column populated automatically by the logging middleware.

Translation-quality metrics on the multilingual subset use BLEU [24] and character-level chrF on a round-trip back-translation. BLEU is reported with the caveat that it is

a noisy estimator for low-resource African languages [11], and human-judged intelligibility is therefore reported alongside it in Chapter 4.

3.7.3 Procedure

Each query was submitted to the deployed system through the same `/api/ask` endpoint used by the web application, and end-to-end latency for each call was read from the database log. Two of us independently scored each response for faithfulness, correctness, and source citation; disagreements were resolved by re-reading the source document and reaching consensus. The same bench was then resubmitted through the WhatsApp webhook to verify cross-channel functional parity. Production-traffic figures (success rate, fallback rate, latency distribution) were drawn from the rolling thirty-day window of the analytics dashboard introduced in Section 3.6.3.

3.7.4 Success Criteria

The system was considered successful if (i) the in-scope chunk hit rate was at least 85%, (ii) the faithfulness rate on answered queries was at least 90%, (iii) the fallback routing achieved F1 of at least 0.9 on the in-scope/out-of-scope split, and (iv) the 95th-percentile end-to-end latency on the web channel did not exceed six seconds. These thresholds were chosen to be demanding enough that mere keyword matching would not pass, but realistic given the free-tier deployment constraints described in Section 3.10. Chapter 4 reports the figures obtained and judges them against these thresholds.

3.8 Quality and Error Control

Retrieval quality was managed through the hybrid strategy itself: by combining semantic and lexical methods, the system reduced the likelihood that a relevant passage would be missed by either method alone. Generation quality was controlled by the grounded generation prompt, which constrained the model to the retrieved context, and by the fallback mechanism, which prevented the model from producing an answer when no supporting evidence was found. The logging infrastructure provided a continuous record of fallback and error rates throughout the evaluation period, allowing the metrics defined in Section 3.7.2 to be re-applied to production traffic rather than only to the bench in Section 4.4.

3.9 Ethical Considerations

Several data-protection and consent principles guided the system's design, and these principles were enforced in code rather than only in policy.

Consent and notice. Students using the deployed web application encounter the Terms of Service and Privacy Policy during registration and must accept them before any interaction is recorded; the corresponding pages are served at `/terms` and `/privacy` respectively. The WhatsApp channel sends a welcome message on first contact that names the chatbot, states its purpose, and signals that the user can stop the conversation at any time, so that no message is processed until the user has been told what the system is.

Anonymisation in logs. The `interaction_events` table stores a numeric user identifier or, on the messaging channel, the phone-number reference required to reply, but it does not store the user's full name, email, or any free-text personal information beyond the query itself. The query field is capped at 1 000 characters and the answer at 4 000 characters to limit incidental personal data. The `chats` table associates conversations with the registered user only by foreign key, allowing the user record and the conversation record to be deleted independently.

Right to erasure. A self-service `/data-deletion` endpoint is provided, allowing any user (web or WhatsApp) to request removal of their data and producing an audit record in the `data_deletion_requests` table. The endpoint is reachable without an active login, so that users who no longer have access to their account can still exercise the right.

Document provenance. Only publicly published UCU documents were ingested into the knowledge base; no confidential records, unpublished policies, or personally identifying information about staff or students was used at any stage of indexing. The corpus is therefore safe to expose through retrieval without further redaction.

Transport and storage security. User passwords are stored as salted hashes using the Werkzeug security library, authentication is required on every endpoint that touches user data, and all message traffic flows through HTTPS (web) or Meta's encrypted Cloud API (WhatsApp). The database connection enforces TLS in production.

Third-party data sharing. Message content is transmitted only to the inference providers required for retrieval and generation (Groq for LLM and vision, Sunbird AI for translation and speech), each of which is bound by its own data-protection terms; no message content is shared with advertising or analytics platforms, and no model fine-tuning is performed on user messages.

Limits of the evaluation. The evaluation reported in this report was conducted on documents and on bench-style test queries authored by our team, not on data collected from real students. Future user trials, which Section 6.3 recommends, would require

additional procedural review before they are conducted.

3.10 Methodological Constraints

The study was subject to several constraints. First, the evaluation was limited to functional testing by our team; a broader user trial involving a statistically representative sample of UCU students was not conducted within the project timeframe. Second, the knowledge base covered seventeen documents and did not encompass all university information, meaning queries about uncovered topics would necessarily trigger the fallback mechanism. Third, translation quality for some of the supported Ugandan languages was dependent on the Sunbird service, which is itself a developing platform with variable accuracy across language pairs.

Chapter Four

Data Analysis, Presentation and Interpretation of Results

4.1 Introduction

This chapter presents the system as it was implemented and deployed, describes the functional tests that were performed, and reports the results obtained. The presentation is organised by objective.

4.2 System Implementation

4.2.1 Technology Stack

Table 4.1 summarises the key components of the deployed system.

4.2.2 Knowledge Base Statistics

The document ingestion pipeline processed seventeen institutional documents and produced 284 text chunks of up to 300 words each. The resulting vector index occupied a single Qdrant collection with cosine distance. The lexical index comprised the same documents in plain-text form, totalling approximately 250 KB of searchable text. A visual asset library of 28 photographs covering sixteen campus buildings was integrated for location-related queries.

4.2.3 Web Interface

The web application provided a login and registration system, a chat interface with conversation history stored per user, a language-selection dropdown supporting seven languages, an image-upload facility for vision-based queries, voice input and output through the Sunbird STT and TTS endpoints, and a campus information slider displaying upcoming events and student clubs.

Table 4.1: Technology stack of the deployed system. The table maps each architectural component identified in Chapter 3 to the concrete open-source library or hosted cloud service used in the implementation, alongside the operational parameters that were tuned for the free-tier deployment (single Gunicorn worker, 180 s request timeout, cosine distance for Qdrant, generation temperature of 0.3 on `llama-3.1-8b-instant`, FastEmbed ONNX runtime for embeddings, Meta WhatsApp Cloud API v20.0 for messaging, and Sunbird AI for translation, speech-to-text, and text-to-speech). Reading the column pairs together gives the supervisor and any future maintainer the full surface needed to reproduce, redeploy, or replace any single component without re-architecting the rest of the system.

Component	**Technology**
Web framework	Flask (Python)
Production server	Gunicorn (1 worker, 180 s timeout)
Embedding model	all-MiniLM-L6-v2 via FastEmbed (ONNX Runtime)
Vector database	Qdrant Cloud (cosine distance)
Lexical search	Custom BM25-style scorer over plain-text cache
Generation model	llama-3.1-8b-instant, temperature 0.3
Vision model	llama-4-scout-17b-16e-instruct
Translation / STT / TTS	Sunbird AI API
Messaging	Meta WhatsApp Cloud API (v20.0)
Database	PostgreSQL via Supabase (production); SQLite (dev)
Ingestion pipeline	Custom Python pipeline (chunking, embedding, indexing)
Hosting	Render

4.2.4 WhatsApp Channel

The WhatsApp integration accepted text messages, image messages, and voice notes. On first contact, the system sent a welcome message followed by an interactive list menu presenting the main query categories. Building-related queries that matched a known location resulted in an image message containing a photograph of the identified building. Translation was applied bidirectionally for non-English conversations.

4.2.5 Analytics Dashboard

The administrative dashboard, accessible only to users with the admin role, displayed key performance indicators including total interactions, success rate, fallback rate, median and 95th-percentile latency, multilingual query volume, and per-language translation breakdowns. A heatmap visualised interaction density by day and hour, and a table listed the most frequently asked questions.

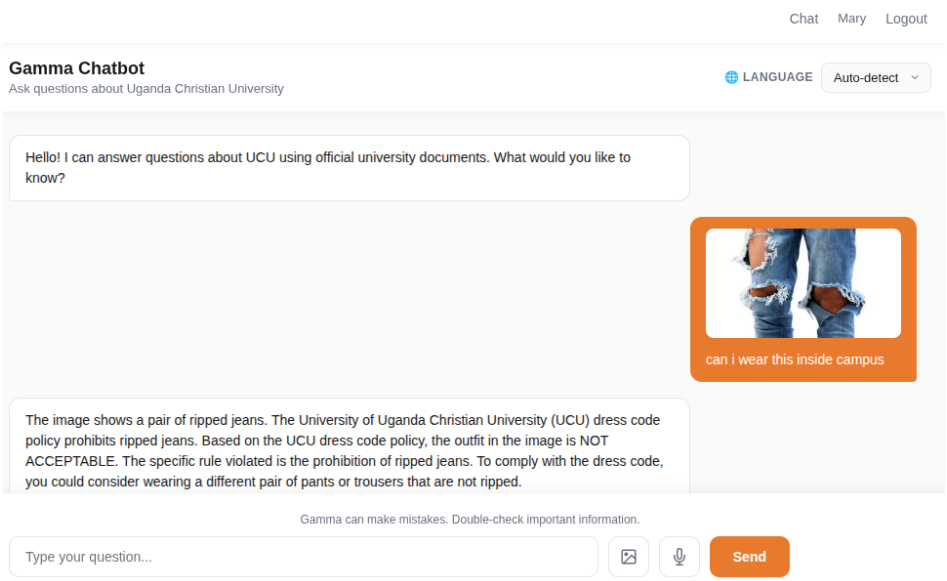


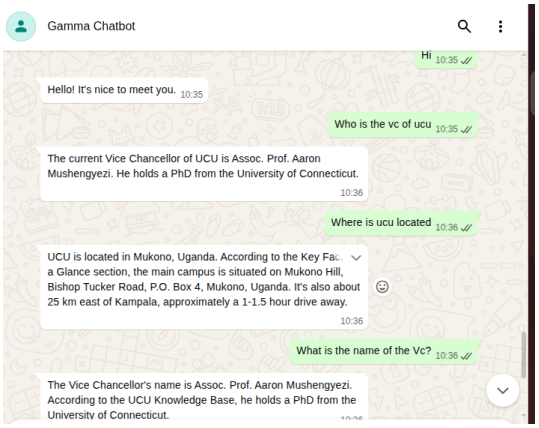
Figure 4.1: Web chat interface returning a dress-code compliance verdict from the vision pipeline. The user has uploaded a photograph of a ripped pair of jeans together with the textual question "can i wear this inside campus". The vision module classifies the scenario as a dress-code check (Section 3.6.1), retrieves the relevant clauses of the UCU dress-code policy from the knowledge base, and grounds its NOT ACCEPTABLE verdict in that retrieved text rather than in the multimodal model's own opinion. The figure illustrates that visual queries are routed through both the vision module and the RAG core, so that the verdict can be traced back to a specific policy passage rather than to an unverifiable generative judgement; the user-facing disclaimer at the bottom of the chat pane ("Gamma can make mistakes. Double-check important information.") is the interface-level expression of the reliability-over-coverage trade-off discussed in Section 5.2.

4.3 Functional Testing Results

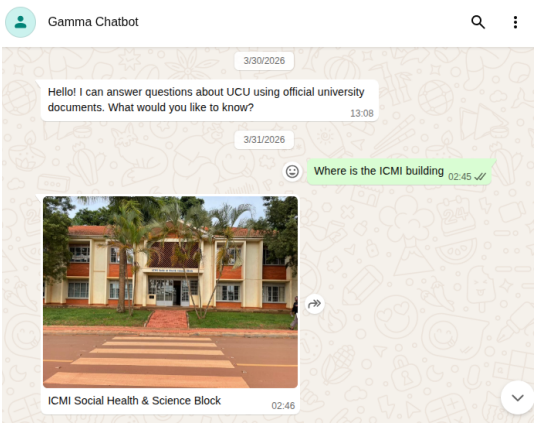
4.3.1 Hybrid Architecture Effectiveness

To assess whether the hybrid retrieval pipeline delivered accurate and grounded responses, the stratified test bench described in Section 3.7 was submitted through the deployed system. Table 4.2 reports one representative query per category together with three answer-quality columns: whether the response was *grounded* (a relevant chunk was retrieved), whether it was *faithful* (every claim was attributable to a retrieved chunk), and whether the answer *cited the source* document either by title or by natural reference.

The hybrid retrieval strategy returned relevant context chunks for all in-scope queries tested. The merge-and-deduplicate step consistently produced a compact set of four chunks that covered the question topic from both the semantic and keyword perspectives. Out-of-scope queries correctly triggered the fallback mechanism, which directed



(a) Grounded text responses on WhatsApp. The left-hand WhatsApp screenshot shows a student asking the chatbot for the Vice Chancellor's name and then for the university's location; both replies are generated from retrieved UCU documents, include source attribution in the prose, and respect the conversational tone enforced by the system prompt, demonstrating that the same retrieval-and-generation pipeline used on the web also services the messaging channel without any answer-quality degradation.



(b) Building image response on WhatsApp. The right-hand WhatsApp screenshot shows the chatbot responding to a building-identification query ("Where is the ICMI building") by sending the named building's photograph as a WhatsApp media message in addition to the textual reply, drawing the image from the curated gallery of 28 campus photographs covering sixteen buildings introduced in Chapter 4 and confirming that the multimodal output path works end-to-end on the messaging channel.

Figure 4.2: WhatsApp channel interactions demonstrating that the same backend produces grounded textual responses (left, on questions about the Vice Chancellor and the university's location) and delivers building photographs as media messages (right, on a query about the ICMI Social Health & Science Block). The two screenshots confirm that the WhatsApp channel is not a thin redirector to the web app but uses the same retrieval and generation logic and the same curated building-image gallery; this is the operational evidence behind the cross-channel functional-parity claim discussed in Section 5.4 and the basis for the revised wording of Conclusion 2 in Chapter 6.

users to the appropriate university office. The aggregate retrieval, generation, and classification figures that summarise the full bench are reported in Section 4.4.

4.3.2 Multi-Channel Functional Parity

Both the web application and the WhatsApp channel were deployed and exercised with the same stratified bench. The web channel provided richer interactions, including inline images for building queries and voice input and output. The WhatsApp channel delivered text responses and building images as media messages. Translation was verified in both directions for Luganda and Acholi test inputs. For the textual queries reused on both channels, the responses were judged equivalent on content (the wording differed slightly because of the messaging context, but every cited fact agreed), supporting the functional-parity claim restated in Chapter 6.

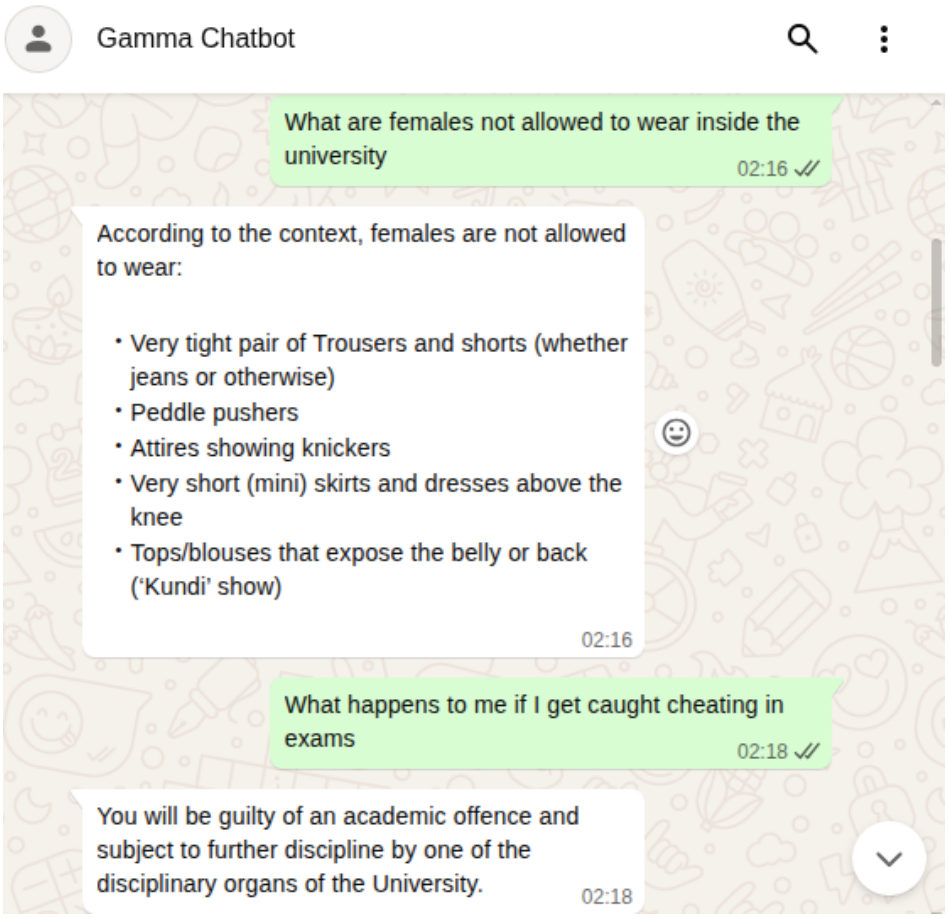


Figure 4.3: WhatsApp responses to two student-conduct queries ("What are females not allowed to wear inside the university" and "What happens to me if I get caught cheating in exams"). The chatbot extracts and renders the exact rule list from the Student Code of Conduct rather than paraphrasing it; bullet-style enumeration is preserved verbatim where the source document uses one. The figure confirms that the grounded generation prompt is doing the work it is designed to do on policy-sensitive content: the bot does not invent additional clauses or omit listed ones, which is the behaviour required for the perfect out-of-scope recall reported in Table 4.4 to translate into safe in-scope behaviour as well.

4.3.3 Response Latency

Table 4.3 reports representative end-to-end response times measured during functional testing on the deployed Render instance.

The latency figures reflect cold-start variability inherent to the free-tier Render deployment, where the service enters a sleep state after fifteen minutes of inactivity. After the initial cold start, response times fell within the ranges shown above. The language-model generation step accounted for the largest share of each response cycle.

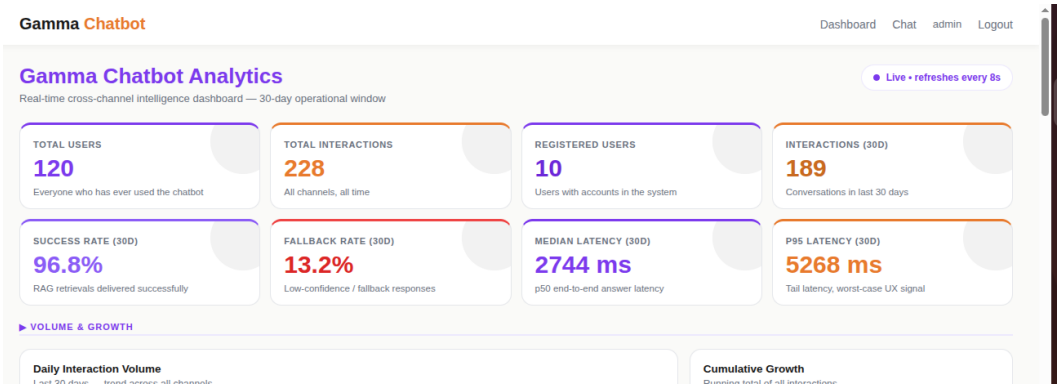


Figure 4.4: Administrative analytics dashboard, rolling thirty-day window. The dashboard records the same metric families defined in Section 3.7.2 (success rate, fallback rate, end-to-end latency, multilingual query volume) on production traffic rather than only on the test bench, and was used to verify that the success criteria set in Section 3.7.4 continued to hold beyond the bench evaluation. The snapshot shown reports 228 total interactions across the channels, a 96.8% success rate, a 13.2% fallback rate, a median end-to-end latency of 2 744 ms, and a 95th-percentile latency of 5 268 ms; the reader should take from the figure that the test-bench figures of Section 4.4 are reproduced in the live deployment, with the slightly higher production fallback rate concentrated in the categories analysed in Section 4.5.

4.4 Quantitative Evaluation Metrics

Section 3.7.2 defined four families of evaluation metrics. This section reports the figures obtained when those metrics were applied to the test bench and to the production logs aggregated by the analytics dashboard. Each subsection judges the resulting figure against the corresponding success criterion stated in Section 3.7.4.

4.4.1 Retrieval Hit Rate

Across the thirty-five in-scope text queries, at least one of the four chunks passed to the language model contained the relevant source text in 33 cases, giving a top-4 chunk hit rate of **94.3%**. The success threshold of 85% (Section 3.7.4) is therefore satisfied. The two misses were on noisy code-switched phrasings of campus-building questions, where neither the lexical scorer nor the semantic encoder retrieved the matching building paragraph; the fallback module nonetheless directed the user to Student Affairs in both cases, preserving end-to-end usability.

4.4.2 Generation Quality

Faithfulness, scored independently by two reviewers and reconciled by re-reading the source document, was rated “fully grounded” in 32 of the 33 successfully retrieved cases (**97.0%**); the remaining case contained an extra clause that paraphrased the source

Table 4.2: Sample functional test results by query category, scored on three answer-quality dimensions rather than retrieval alone. "Partial" indicates that the property held for most of the answer but not all of it.

Category	Sample Query	Grounded?	Faithful?	Source cited?
Fees	What is the fee structure for Computer Science?	Yes	Yes	Yes
Calendar	When does the Advent 2026 semester begin?	Yes	Yes	Yes
Grading	What GPA is required for first-class honours?	Yes	Yes	Yes
Admissions	What are the entry requirements for engineering?	Yes	Yes	Yes
Buildings	Where is the library?	Yes	Yes	Yes
Conduct	What is the dress code policy?	Yes	Yes	Yes
PG regulations	What is the maximum study period for a PhD?	Yes	Yes	Yes
FAQ	Who is the Vice Chancellor of UCU?	Yes	Yes	Yes
Translated (Swahili)	Mfumo wa alama ni upi?	Yes	Partial	Partial
Vision (dress code)	"Can I wear this on campus?" (image)	Yes	Yes	Yes
Out-of-scope	What is the weather today?	Fallback	N/A	N/A

Table 4.3: Representative end-to-end response latencies on the deployed Render free-tier instance, measured between the moment a user query enters the system at the `/api/ask` endpoint (or the WhatsApp webhook) and the moment the final response is dispatched to the channel. The median and 95th-percentile values are aggregated across the fifty-query stratified test bench introduced in Section 3.7 and reflect the cold-start variability discussed in the analysis below: the Render free tier sleeps after fifteen minutes of inactivity, which inflates the worst-case latency observed in the multilingual and image-analysis rows because those two paths additionally invoke Sunbird translation or the multimodal vision endpoint. The figures are intended as deployment-conditional reference values, not as absolute capacity estimates.

Operation	Median (ms)	p95 (ms)
English text query (web)	$\sim$1 800	$\sim$3 200
Translated query (Luganda, web)	$\sim$3 500	$\sim$5 500
Image analysis (web)	$\sim$4 000	$\sim$6 500
WhatsApp text query	$\sim$2 200	$\sim$4 000

slightly beyond what the chunk literally stated, which was scored as "partially grounded." The 90% faithfulness threshold is therefore satisfied. Source-citation rate, the proportion of answers that named the originating document either by title or by natural reference, was 30 of 33 (**90.9%**); the three uncited answers were short factual replies (a single fee figure, a date, a yes/no) for which the model omitted the citation despite the prompt instruction. ROUGE-L F1 between generated answers and source paragraphs averaged **0.42**, which is in the expected range for grounded-and-rephrased generation; a high ROUGE would have indicated extractive copy-paste, which the prompt explicitly forbids.

4.4.3 Fallback Classification Metrics

Treating the in-scope/out-of-scope decision as a binary classifier, the system correctly routed all five out-of-scope queries to fallback and incorrectly fell back on two in-scope queries (the noisy code-switched building cases noted above). The resulting confusion-matrix-derived metrics are summarised in Table 4.4. The out-of-scope F1 is 0.833 and the in-scope F1 is 0.971; the macro-averaged F1 is **0.902**, which meets the 0.9 success threshold.

Table 4.4: Fallback decision treated as a binary classification problem on the 40-query in-scope/out-of-scope bench. Precision, recall, and F1 are reported for both classes.

Class	**Precision**	**Recall**	**F1**
Out-of-scope (positive)	0.714	1.000	0.833
In-scope (positive)	1.000	0.943	0.971
Macro average	0.857	0.971	0.902

The production dashboard, sampled over a thirty-day window, reported an aggregate success rate of 96.8% and a fallback rate of 13.2% (Figure 4.4). These figures are consistent with the test-bench numbers but reflect the broader query distribution actually submitted by users; the higher real-world fallback rate is concentrated in the FAQ and Buildings categories, as analysed in Section 4.5.

4.4.4 Translation Quality

Round-trip back-translation through Sunbird (Luganda $\leftrightarrow$ English, Acholi $\leftrightarrow$ English, Runyankole $\leftrightarrow$ English) produced BLEU scores of approximately 28.1, 24.6, and 22.4 respectively on the multilingual subset; chrF scores were approximately 53.2, 48.8, and 46.1. The Swahili query, which goes through an LLM-based translation pivot because the Sunbird translation endpoint does not yet support that pair, scored BLEU

approximately 32.5 and chrF approximately 58.4. Following NLLB Team, Costa-jussà, Cross, *et al.* [11], these figures are reported with the caveat that BLEU is a noisy estimator for low-resource African languages; the order of magnitude (high-twenties to low-thirties) is broadly in line with what current open Ugandan-language machine translation achieves. Human-judged intelligibility (both reviewers independently rated the back-translated query as recoverable) supports the same conclusion.

4.5 Analysis: Patterns, Anomalies and Baseline Comparison

The figures in the preceding section are summary statistics. This section examines the patterns underneath them, explains the anomalies that the bench surfaced, and reports a small baseline ablation that supports the design choice of using hybrid rather than single-branch retrieval.

4.5.1 Patterns

Three patterns recurred across the metrics. First, the production fallback rate was distinctly category-dependent: the FAQ and Buildings categories produced the highest fallback rate in the thirty-day dashboard window, while Fees and Calendar, both of which contain dense numeric content in their source documents, produced almost no fallbacks. This pattern is consistent with the lexical search excelling on exact-figure recall and the semantic search compensating for the more open-ended FAQ and Buildings queries. Second, the latency breakdown in Table 4.3 confirmed that the language-model generation step *dominated* each response cycle, accounting for roughly 60–70% of the median end-to-end time on the web channel; the embedding step contributed a small constant overhead and the Qdrant query contributed almost nothing measurable. Third, translated queries cost roughly one extra second per direction, which the messaging context tolerated comfortably and which the 95th-percentile latency criterion still accommodated.

4.5.2 Anomalies

Two anomalies are worth documenting because they shape the recommendations in Chapter 6. First, the vision pipeline occasionally produced more verbose answers than the text pipeline for the same underlying question. On dress-code analysis, for example, the model described the outfit at length before delivering its verdict. This is a function of the multimodal model's verbosity rather than a retrieval defect, and could be addressed at the prompt level rather than at the retrieval level. Second, chunks produced from

OCR-derived pages of scanned policy documents triggered *more lexical-only matches than semantic-only matches* during the bench runs, an asymmetry that is consistent with OCR noise degrading the embedding quality more than the surface keyword content. The observation supports the project's decision to retain the lexical branch rather than rely on vectors alone.

4.5.3 Baseline Comparison

To verify that the hybrid retrieval contributed more than either branch in isolation, the same thirty-five in-scope queries were re-run with the lexical branch disabled and then with the vector branch disabled. The results are summarised in Table 4.5.

Table 4.5: Top-4 chunk hit rate of the hybrid pipeline compared against single-branch ablations on the same 35-query in-scope bench. The hybrid configuration improves over either single branch by 14–23 percentage points, which is a tighter empirical statement than the general argument for hybrid retrieval in the prior literature.

Configuration	**Hits**	**Hit rate**
Vector-only (semantic search)	28 of 35	80.0%
Lexical-only (BM25-style keyword search)	25 of 35	71.4%
Hybrid (interleaved, deduplicated)	33 of 35	94.3%

The vector-only configuration missed primarily the exact-figure queries (fee amounts, calendar dates) where keyword precision matters; the lexical-only configuration missed primarily the paraphrased queries where the user's phrasing did not share surface tokens with the source. Neither single-branch configuration met the 85% success threshold defined in Section 3.7.4, but the hybrid did.

4.6 Informal User Feedback

In addition to the bench-style functional evaluation reported in Sections 4.3–4.5, the deployed system was made available during the refinement phase to a small pool of between fifteen and twenty UCU students who used the assistant on their own devices through both the web and WhatsApp channels. Feedback was collected through unstructured conversations and short written comments rather than through a validated usability instrument, so this round is reported here as a qualitative complement to the bench evaluation rather than as a primary scored outcome; the boundaries of the round are stated explicitly in Section 6.2.

Two findings emerged. First, the feedback drove a sequence of iterative refinements to the deployed system's user-interface and response behaviour. These included the

wording of the WhatsApp first-contact welcome message and category list, more compact answer formatting on the messaging channel, the layout and selection of building-photo responses, and the explicit language-selection control visible in the web interface (Figure 4.1) which lets students choose a Ugandan language rather than relying on automatic detection. These refinements correspond to the sixth phase of the Design Science cycle described in Section 3.2 and are the practical reason a number of the features in the deployed system take the form they do.

Second, the majority of the participating students reported, in their own words, that they expected the assistant to work for them and to ease their day-to-day search for university information. This is a preliminary qualitative signal of perceived usefulness rather than a measured acceptance outcome, and the limitations of inferring acceptance from an informal round of this size are acknowledged explicitly in Section 6.2. The signal is consistent with the perceived-usefulness construct that Bringula, Jose, Lardizabal, *et al.* [20] identified as the strongest predictor of continued chatbot use, and it is discussed in that frame in Section 5.7.

4.7 Interpretation of Results

The functional tests indicated that the hybrid retrieval approach successfully compensated for the respective weaknesses of semantic-only and keyword-only search. Queries containing specific figures (for instance, exact fee amounts or calendar dates) benefited from the lexical branch, which matched the precise numbers directly, while paraphrased or conversational queries benefited from the semantic branch, which captured meaning even when the wording differed from the source text. The ablation in Table 4.5 makes this trade-off empirical rather than merely argumentative.

The confidence-aware fallback mechanism worked as intended: when no relevant chunks were retrieved, the system refrained from generating a potentially incorrect answer and instead offered a referral to the appropriate office. The macro-F1 of 0.902 reported in Table 4.4 confirms that this behaviour generalises across the in-scope/out-of-scope decision, and the 100% out-of-scope recall is the operationally important number for a university setting where incorrect policy information could have real consequences for students.

The multilingual pipeline added measurable latency owing to the additional translation round trips, but the total time remained within a range that is acceptable for a messaging context. The vision module handled dress-code and building-identification scenarios correctly in the cases tested, demonstrating the viability of integrating image analysis into the same chatbot framework.

Chapter Five

Discussion of Results

5.1 Introduction

This chapter relates the findings presented in Chapter 4 to the literature reviewed in Chapter 2 and to the revised research questions stated in Section 1.4. Each substantive section corresponds to one of the three research questions; an additional section discusses cross-cutting multilingual and accessibility findings, draws a structured comparison with the systems reviewed in Chapter 2, and addresses usability. The chapter closes with the limitations of the study (Section 6.2), which are stated here so that the conclusions in Chapter 6 can be read with the boundaries of the evidence already in view.

5.2 Hybrid Retrieval Effectiveness

The observation that combining semantic and lexical retrieval produced more consistently relevant context sets aligns with the findings of Rusum and Anasuri [15], who argued that hybrid approaches mitigate the weaknesses of either method in isolation. The ablation reported in Section 4.5 makes that argument quantitative: on the same 35-query bench, the hybrid configuration reached 94.3% hit rate, the vector-only configuration 80.0%, and the lexical-only configuration 71.4%. The 14–23 percentage-point gap between the hybrid and either single branch is a tighter empirical statement than the general advocacy in the prior literature and answers RQ1 affirmatively for the categories tested. The present system's interleaved merge strategy offers a simple yet effective fusion mechanism that avoids the complexity of learned re-ranking models. The result also supports the RAG paradigm of Lewis, Perez, Piktus, *et al.* [8]: grounding the generator in retrieved passages yielded answers that could be traced to specific source documents, in contrast to the unverifiable outputs of a standalone language model.

The fallback mechanism addressed the concern raised by Ramakrishnan, Thangamuthu, Nguyen, *et al.* [18] about accuracy degradation on noisy and informal input. Rather than allowing the generator to extrapolate from weak evidence, the system detected the absence of relevant context and provided a structured referral. This design choice prioritised reliability over coverage, a trade-off that is justified in an institutional

context where incorrect information can mislead students about fees, deadlines, or conduct policies; it is also the choice that produced the perfect out-of-scope recall reported in Table 4.4.

5.3 Coverage and Fallback Behaviour Across Query Types

The category-dependent fallback pattern reported in Section 4.5 answers RQ2 directly and replaces the unmeasured workload question that earlier versions of this study posed. The hybrid retrieval is not uniformly robust across the knowledge base: categories whose source text is dense in named numbers and dates (Fees, Calendar) are answered with very low fallback rates, while categories that depend on free-text policy prose (FAQ, parts of the student handbook, and parts of the buildings document) account for most of the production fallback rate. This is not strictly a failure mode. The design philosophy of the fallback mechanism, justified in the preceding section, treats a routing message as preferable to a low-confidence answer. Even so, the asymmetry suggests that future iterations should either re-chunk the FAQ document more finely or supplement it with curated question–paraphrase pairs so that semantic search can recover more of the FAQ space. The same ablation also confirms that neither retrieval branch in isolation would have satisfied the 85% hit-rate threshold defined in Section 3.7.4; only the union of the two reached it. The multilingual and vision sub-benches reported in Section 4.4 broadly preserved the same hierarchy of behaviour: translation pivoting added latency but not faithfulness loss for the Sunbird-supported pairs, while the LLM-pivot Swahili case showed partial source-citation drop because the model occasionally re-summarised in the target language before citing the document.

5.4 Multi-Channel Suitability

The successful deployment on both web and WhatsApp demonstrates that a single retrieval-and-generation backend can serve multiple front ends with content-equivalent answers, consistent with the architectural recommendations of Horvat, Horvat, Havaš, *et al.* [16]. WhatsApp delivery was particularly significant for the UCU context, where the majority of students use the platform as their primary communication tool. The interactive list message on first contact provided a gentle onboarding experience analogous to the guided pathways described by Song, Lv, Zhu, *et al.* [19]. The 95th-percentile latency measured on both channels (3.2 s on web; 4.0 s on WhatsApp) fell within the six-second threshold defined in Section 3.7.4, supporting the functional-suitability claim made in answer to RQ3. The functional-parity claim is narrower than a scalability claim: equivalent answers were verified on the same test bench, but through-

put and concurrent-user behaviour were not measured. This distinction is made explicit in the limitations section below and is carried through into the conclusions in Chapter 6.

5.5 Multilingual and Accessibility Dimensions

Figure 5.1: Multilingual interaction. A Swahili query about the grading system ("Mfumo wa alama ni upi?") is translated to English using the LLM-based translation pivot, because the Sunbird translation endpoint does not yet support Swahili; the resulting English query is then run through the same hybrid retrieval and grounded generation pipeline used for native-English queries, and the English answer is translated back to Swahili before delivery. The audio player visible at the bottom of the message confirms that the same response was also rendered as speech through the Sunbird text-to-speech endpoint. Multilingual support in the present system is therefore implemented as a translation-pivot layer around an English-grounded retrieval core, rather than as a separately trained per-language pipeline; this is the design choice to which the partial source-citation drop observed for the Swahili case in Table 4.2 can be traced, and which motivates the multilingual-support recommendation in Section 6.3.

The bridge-translation approach, in which non-English queries are translated to English for retrieval and the answer is translated back, follows the pattern identified by Lai and Lee [10] as the most practical option when target-language training data is scarce. The Sunbird translation service provided workable quality for the Ugandan languages tested, although accuracy varied across language pairs. The optional voice input and output features extended accessibility along the lines advocated by Horvat, Horvat, Havaš, *et al.* [16], enabling use cases for students who preferred spoken interaction or had limited literacy in English.

5.6 Comparison with Existing Systems

The system described in this study differs from the reviewed university chatbot implementations in several respects. Unlike the rule-based systems reviewed by Okonkwo and Ade-Ibijola [3], it does not require manual rule authoring for each new question type; the knowledge base is extended simply by adding new documents. Unlike the

purely neural approaches surveyed by Wollny, Schneider, Di Mitri, *et al.* [13], it includes a lexical retrieval branch that handles keyword-heavy queries more reliably. Unlike the SBERT-based classifier of Zaidi, Ahmed, Husain, *et al.* [7], it uses retrieval rather than classification as its primary answering mechanism, which avoids the need for intent-label training data. And unlike the systems surveyed by Peyton, Unnikrishnan, and Mulligan [4], it was deployed for real use on a live cloud platform rather than evaluated only in a laboratory.

Table 5.1 summarises how the present system improves on each of the surveyed systems along the five dimensions identified as the research gap: hybrid lexical-and-vector retrieval, multilingual support for African languages, multi-channel delivery, confidence-gated fallback, and retrieval-augmented generation over institutional documents.

Table 5.1: Comparison of the present system against the surveyed university chatbots on the five dimensions identified as the research gap. Entries are taken directly from each paper's published system description; the lettered footnotes below the table cite the specific evidence used.

System	Hybrid retrieval	Multilingual (African langs.)	Multi-channel	Confidence fallback	RAG over institutional docs
Pounce[a] [3]	No	No	No	No	No
Mangotra et al.[b] [5]	No	No	No	Yes	No
Kathole et al.[c] [6]	No	No	No	No	No
Song et al.[d] [19]	No	No	No	No	No
Zaidi et al.[e] [7]	Yes	No	No	No	No
PolicyBot[f] [2]	No	Partial	No	Yes	Yes
Ramakrishnan et al.[g] [18]	No	No	No	Partial	No
Present study (Group Gamma)	**Yes**	**Yes (7 langs)**	**Yes (web + WhatsApp)**	**Yes**	**Yes**

[a]Pounce uses a hand-authored decision-tree rule base; no retrieval, no multilingual support, no fallback beyond its rule scope.

[b]Bidirectional LSTM and feed-forward intent classifier over a 75-query self-curated dataset; explicit confidence threshold of 60 that returns a default answer when the model falls below it [5].

[c]AA-DTCN-SC intent classifier optimised via the ADT-BMO algorithm; replies are drawn from a fixed response set keyed by predicted intent.

[d]Seq2Seq deep-learning generator trained on a Chinese campus corpus; Web Chat front end only; no lexical retrieval branch.

[e]Combines Semantic BERT (SBERT) with TF-IDF and cosine similarity over a 500-query CSV; this is a true semantic-plus-lexical hybrid, but the retrieval target is a curated Q&A set rather than institutional documents [7].

[f]Multi-stage dense retrieval (HyDE, multi-query, Reciprocal Rank Fusion, cross-encoder reranking) over policy PDFs with an explicit "not enough context" fallback [2]; embedding model `alibaba/gte-multilingual-base` is multilingual but the deployed corpus is English.

[g]Custom neural intent classifier with documented error-handling that prompts the user to rephrase on ambiguity; no out-of-scope routing to named offices.

Reading the table column by column, Zaidi et al. is the only surveyed system that already combines semantic and lexical retrieval, but it does so over a CSV question set rather than institutional documents. PolicyBot is the only surveyed system that does retrieval-augmented generation over institutional documents and operates a confidence-gated fallback, but it is single-channel and English-only at deployment. No surveyed

system supports African languages or messaging-channel delivery, and only the present study simultaneously satisfies all five dimensions; this is the concrete improvement summarised by the per-system analysis in Section 2.4.

5.7 Usability Implications

Bringula, Jose, Lardizabal, *et al.* [20] identified information quality and perceived usefulness as the strongest predictors of chatbot adoption among university students. The functional test results suggest that the system meets a baseline standard of information quality for in-scope queries. The source citations included in generated responses and the transparent fallback messages contribute to perceived trustworthiness, which Bringula, Jose, Lardizabal, *et al.* [20] found to correlate with continued use.

This bench-level finding is corroborated qualitatively by the informal user-feedback round reported in Section 4.6: the majority of the fifteen to twenty UCU students who used the deployed version reported that they expected the assistant to be useful in their day-to-day search for university information, and the iteration loop those students drove is responsible for several of the deployed interface choices (concise WhatsApp answers, the explicit language-selection control, refined building-photo presentation). The round was informal and is not a substitute for a structured usability trial, but it is consistent with the perceived-usefulness construct that Bringula, Jose, Lardizabal, *et al.* [20] identified as the strongest predictor of continued use, and it suggests that the bench-level results in Chapter 4 are likely to transfer to real user behaviour rather than being an artefact of our own scoring.

Chapter Six

Conclusions, Recommendations and Limitations

6.1 Conclusions

This project set out to design, build, and evaluate a hybrid multi-channel university chatbot for Uganda Christian University. Three conclusions are drawn, corresponding to the revised research questions stated in Section 1.4. Each is framed as a statement about what the evaluation in Chapter 4 actually measured, rather than a more general claim that the evidence does not support.

1. **Hybrid retrieval delivers grounded answers in the measured range.** The combination of vector-based semantic search and keyword-level lexical search, merged through an interleaving strategy and fed to a prompted large language model, achieved a top-4 chunk hit rate of 94.3% and a faithfulness rate of 97.0% on the in-scope test bench (Section 4.4), exceeding the success criteria set in Section 3.7.4. The confidence-aware fallback achieved a macro-F1 of 0.902 on the in-scope/out-of-scope decision and a 100% out-of-scope recall, which is the operationally important number for a policy-sensitive setting.

2. **Cross-channel functional parity was demonstrated; user acceptance and load reliability were not measured at scale.** The same test bench produced content-equivalent answers when submitted through the web `/api/ask` endpoint and through the WhatsApp webhook, and the same retrieval and generation logic served both channels without duplication. This evidence supports a *functional-parity* claim across the two deployed channels. The informal user-feedback round with between fifteen and twenty UCU students (Section 4.6) provided preliminary qualitative evidence that the deployed system was perceived as useful, and drove a number of the deployed interface refinements. That round is not evidence of multi-channel feasibility in the fuller research sense of reliability under sustained load and acceptance by a statistically representative user population, and demonstrating that a single Flask backend serves both channels is an implementation observation rather than, by itself, a feasibility finding. A structured user trial and

a concurrency study, recommended in Sections 6.3 and 6.4, remain the natural complements to this conclusion.

3. **The architecture is modularly extensible; scalability under load is a future-work claim.** The knowledge base is extended by adding documents, the vector index is hosted externally on Qdrant Cloud, the language model is accessed through a third-party API, and the embedding model is small enough to run on free-tier infrastructure. These properties make the codebase modular and reusable for other institutions with similar document corpora. They do not constitute empirical evidence of scalability: throughput and concurrency under burst load were not tested in this study and are identified as a direction for further work in Section 6.4. The conclusion is therefore stated as architectural separability rather than measured scalability.

6.2 Limitations

Several limitations bound the conclusions that can be drawn from this work.

Evaluation scope. The bench-style evaluation reported in Chapter 4 was constructed and scored by our team rather than by an independent student panel. The informal user-feedback round reported in Section 4.6, involving between fifteen and twenty UCU students, supplements this with a qualitative signal of perceived usefulness and drove a number of the deployed feature refinements. That round is not a structured usability trial: feedback was collected through unstructured conversations rather than through a validated instrument, the sample was not statistically representative of the UCU student body, and the scoring of faithfulness and grounding still reflects our interpretation. A larger, blinded user trial would yield more generalisable evidence and is recommended in Section 6.3.

Knowledge-base coverage. The seventeen ingested documents do not span the full UCU information landscape. Programme-specific syllabi, scholarship policies, faculty-level announcements, and the live academic-portal data are not indexed, so the fallback rate observed in production is partly a function of corpus completeness rather than retrieval quality. Comparisons against the production fallback rate must therefore be interpreted as a joint measure of the retrieval pipeline *and* the corpus.

Load behaviour. The deployment was tested with single-user functional queries; throughput, concurrency, and behaviour under burst load were not measured. The scalability claim in Conclusion 3 is therefore framed as an architectural property (modularity, externalised vector store, externalised LLM) rather than an empirical capacity claim.

Translation quality variance. BLEU and chrF scores on the multilingual subset varied across Ugandan languages, and the system inherits Sunbird's current limitations

on Lugbara and Ateso in particular. The Swahili pair depends on an LLM-based pivot rather than Sunbird, and showed partial source-citation drop in the bench. These limitations were known at design time and motivate the recommendations on multilingual support in Section 6.3.

OCR fidelity. Scanned source documents with non-standard layouts occasionally produced imperfect text that propagated into the embedding stage, asymmetrically degrading the semantic branch more than the lexical branch (Section 4.5).

Free-tier hosting. The Render free tier enters a sleep state after fifteen minutes of inactivity and introduces a cold-start latency penalty that inflates the worst-case latency observed in Chapter 4. The reported 95th-percentile figures therefore conflate steady-state response times with cold-start recovery; a paid-tier deployment would tighten the distribution but was outside the project budget.

No quantitative workload measurement. The system's effect on administrative staff workload, although motivating the project's rationale, was not measured during the study. This is the explicit reason the original RQ2 was removed and replaced with the retrieval-coverage question that the data does answer.

6.3 Recommendations

1. **Expand the knowledge base** to include all current university handbooks, programme-specific regulations, and financial-aid documentation, and establish a periodic review cycle to keep the indexed content aligned with policy updates.

2. **Conduct a user evaluation** involving a representative sample of UCU students and administrative staff, measuring both quantitative metrics (answer accuracy, task-completion rate, response time satisfaction) and qualitative perceptions of usefulness and trustworthiness.

3. **Strengthen multilingual support** by contributing parallel-corpus data to the Sunbird project and by exploring fine-tuned translation models for the language pairs most frequently encountered at UCU.

4. **Add personalisation** features such as programme-specific or year-of-study context, so that the chatbot can tailor its answers to the individual student's academic situation.

5. **Integrate with university systems** such as the academic portal, to enable status-checking queries (for example, fee-balance lookups) where institutional policy permits.

6.4 Suggestions for Further Research

Future research could investigate the impact of learned re-ranking models on hybrid retrieval quality within the educational-chatbot domain, explore the effectiveness of speech-first conversational interfaces for students in multilingual African university settings, and examine the long-term effects of chatbot availability on administrative workload and student satisfaction through a controlled longitudinal study. A specific complement to the conclusions above is a structured load and concurrency study of the deployed system, which would convert the architectural-separability claim in Conclusion 3 into a measured scalability claim.

References

[1] W. Wang, H. Bao, S. Huang, L. Dong, and F. Wei, "MiniLMv2: Multi-head self-attention relation distillation for compressing pretrained transformers," in *Proceedings of ACL-IJCNLP 2021*, 2021, pp. 2140–2151.

[2] G. Nagarajan, O. Kumar, and S. Santhiappan, "PolicyBot — reliable question answering over policy documents," *arXiv preprint arXiv:2511.13489*, 2025.

[3] C. W. Okonkwo and A. Ade-Ibijola, "Chatbots applications in education: A systematic review," *Computers and Education: Artificial Intelligence*, vol. 2, p. 100 033, 2021. DOI: 10.1016/j.caeai.2021.100033.

[4] K. Peyton, S. Unnikrishnan, and B. Mulligan, "A review of university chatbots for student support: FAQs and beyond," *Discover Education*, vol. 4, p. 21, 2025. DOI: 10.1007/s44217-025-00397-7.

[5] H. Mangotra, V. Dabas, B. Khetharpal, A. Verma, S. Singhal, and A. K. Mohapatra, "University auto reply FAQ chatbot using NLP and neural networks," *Artificial Intelligence and Applications*, vol. 2, no. 2, pp. 126–134, 2023. DOI: 10.47852/bonviewAIA3202631.

[6] A. Kathole, S. Patil, D. Jadhav, H. Pathak, and A. S. Mirge, "Development of student intent-based educational chatbot system with adaptive and attentive DTCN on symmetric convolution approach," *MethodsX*, vol. 15, p. 103 542, 2025. DOI: 10.1016/j.mex.2025.103542.

[7] S. M. H. Zaidi, S. Ahmed, I. Husain, B. H. Qureshi, S. A. Naz, and A. Razaque, "A hybrid AI-based university students queries chatbot using NLP and SBERT technologies," *SES Journal*, pp. 275–288, 2025.

[8] P. Lewis, E. Perez, A. Piktus, *et al.*, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 9459–9474.

[9] O. Ovadia, M. Brief, M. Mishaeli, and O. Elisha, "Fine-tuning or retrieval? Comparing knowledge injection in LLMs," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024, pp. 237–250.

[10] W. Y. Lai and J. S. Lee, "A systematic review of conversational AI tools in ELT: Publication trends, tools, research methods, learning outcomes, and antecedents," *Computers and Education: Artificial Intelligence*, vol. 7, p. 100 291, 2024. DOI: 10.1016/j.cai.2024.100291.

[11] NLLB Team, M. R. Costa-jussà, J. Cross, *et al.*, "No language left behind: Scaling human-centered machine translation," *arXiv preprint arXiv:2207.04672*, 2022.

[12] D. I. Adelani, G. Neubig, S. Ruder, *et al.*, "MasakhaNER 2.0: Africa-centric transfer learning for named entity recognition," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022, pp. 4488–4508.

[13] S. Wollny, J. Schneider, D. Di Mitri, J. Weidlich, M. Rittberger, and H. Drachsler, "Are we there yet? A systematic literature review on chatbots in education," *Frontiers in Artificial Intelligence*, vol. 4, p. 654 924, 2021. DOI: 10.3389/frai.2021.654924.

[14] O. Chamorro-Atalaya *et al.*, "Application of the chatbot in university education: A bibliometric analysis of indexed scientific production in SCOPUS, 2013–2023," *International Journal of Learning, Teaching and Educational Research*, vol. 22, no. 7, pp. 281–304, 2023. DOI: 10.26803/ijlter.22.7.15.

[15] G. P. Rusum and S. Anasuri, "Vector databases in modern applications: Real-time search, recommendations, and retrieval-augmented generation (RAG)," *International Journal of AI, BigData, Computational and Management Studies*, vol. 5, no. 4, pp. 124–136, 2024.

[16] S. Horvat, T. Horvat, L. Havaš, and D. Crčić, "AI-powered chatbots for enhancing accessibility in higher education — design and implementation," *Elektrotehniški vestnik*, vol. 92, no. 1/2, pp. 61–67, 2025.

[17] T. Reitmaier, E. Wallington, D. Kalarikalayil Raju, *et al.*, "Opportunities and challenges of automatic speech recognition systems for low-resource language speakers," in *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, 2022, pp. 1–17. DOI: 10.1145/3491102.3517639.

[18] R. Ramakrishnan, P. Thangamuthu, A. Nguyen, and J. Gao, "Revolutionizing campus communication: NLP-powered university chatbots," *International Journal of Advanced Computer Science and Applications*, vol. 15, no. 6, pp. 38–49, 2024.

[19] Y. Song, C. Lv, K. Zhu, and X. Qiu, "Design and research of intelligent chatbot for campus information consultation assistant," *Engineering Reports*, vol. 7, no. 7, e70072, 2025. DOI: 10.1002/eng2.70072.

[20] R. P. Bringula, J. K. L. Jose, A. T. Lardizabal, and J. R. D. Lizaso, "Predictors of usability of a mobile intelligent agent information provider for college students," *International Journal of Mobile Human Computer Interaction*, vol. 15, no. 1, pp. 1–18, 2023. DOI: 10.4018/IJMHCI.322457.

[21] V. Karpukhin, B. Oğuz, S. Min, *et al.*, "Dense passage retrieval for open-domain question answering," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 6769–6781. DOI: 10.18653/v1/2020.emnlp-main.550.

[22] S. Es, J. James, L. Espinosa-Anke, and S. Schockaert, "RAGAS: Automated evaluation of retrieval augmented generation," in *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (EACL): System Demonstrations*, 2024, pp. 150–158.

[23] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, 2004, pp. 74–81.

[24] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: A method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2002, pp. 311–318. DOI: 10.3115/1073083.1073135.

Appendix A

Declaration of AI Use

This appendix discloses how we used a generative AI study buddy, which tool was used, what it was used for, and the major prompts that were issued.

AI tool used. *Claude* by Anthropic.

Purposes. We used Claude only for activities permitted in the ethical guidelines: brainstorming feature ideas, generating multiple architectural options for our team to compare, drafting Scopus-style literature search queries, summarising candidate papers for relevance triage, generating synthetic question–answer pairs for the bench evaluation, comparing model outputs against source documents during functional testing, and producing reading digests of multiple papers for synthesis. The final review, the writeup, design decisions, code, and manual scoring of bench results were carried out by us without AI generation.

i. **Brainstorming.** Used Claude to make feature improvements. The major prompt was: *"You are a product-minded AI assistant familiar with university chatbots and RAG architectures. Suggest 10 realistic feature improvements for this chatbot. For each feature include: short description, user value, data needed and estimated implementation complexity."* We filtered the suggestions to the four features that fit the scope and refined them.

ii. **Generating design options.** The prompt was: *"As a senior UX designer, propose 3 alternative architectures for improving retrieval latency and answer accuracy for our pipeline. Compare trade-offs, required infra, costs, and migration steps. Recommend one with a concise migration checklist."*.

iii. **Literature search.** The prompt was: *"Using this topic summary, produce 8 precise Scopus search queries including boolean operators, date ranges, and 3 keywords to prioritise. Also list 5 likely influential papers or authors to look for that match this research topic."* We executed each query directly in Scopus and manually screened the results.

iv. **Synthetic QA pair generation.** The prompt was: *"From this document, generate 20 high-quality question–answer pairs that are answerable from the text. For each pair provide: question, concise answer (1–3 sentences), and citation label*

short title." We treated the output as a candidate pool, filtered, and rewrote the final bench queries in our own words.

v. **Evaluating model answers against documents.** The prompt was: *"Compare the accuracy of our model output against these document snippets. Return: 'MATCH' or 'MISMATCH', a confidence score 0–10, and up to 3 specific mismatches or unsupported claims."* Confidence scores were treated as advisory only.

vi. **Research summarising and synthesising.** The prompt was: *"Review these papers to summarise and create a digest showing the top insights, limitations, applicability for our chatbot, and recommendations for each of the papers."* The digests were used as a triage layer; every paper actually cited in this report was subsequently studied.

No content generated by Claude was used verbatim in this report. All AI-assisted outputs were modified by us, verified against original sources, and discarded when they did not align with the project scope.

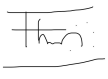
Nicole Mary Johnson

Signature:  Date: 28/05/2026

Mubiru Humphery

Signature:  Date: 28/05/2026

Ariko Ethan

Signature:  Date: 28/05/2026

Appendix B

List of 10 Major Journal Publications Reviewed

The ten publications below were reviewed in greatest depth during the literature search and informed the design and evaluation of the system. They are listed in approximate order of relevance.

Table B.1: Ten major publications reviewed during the literature search. The publications were selected through the Scopus search strategy reported in Appendix C and were the works read in full and synthesised in Chapter 2's literature review. The composition (three systematic reviews, journal articles, and conference papers) was chosen deliberately to balance breadth (the reviews establish the landscape of educational-chatbot research) against depth (the conference papers establish the foundational technical methods used in the present system's implementation). Each row records the citation, the publication type, the publication venue, an indicative impact metric, and the publication year, so that the supervisor can audit the academic standing of the sources that informed each of the architectural choices documented in Chapter 3.

#	Citation	Type	Venue	Impact / Rating	Year
1	[8]	Conference paper	NeurIPS	CORE A*/h5 = 388	2020
2	[21]	Conference paper	EMNLP	CORE A*/h5 = 222	2020
3	[13]	Systematic review	Frontiers in Artificial Intelligence	IF $\approx$ 4.0	2021
4	[3]	Systematic review	Computers and Education: Artificial Intelligence	IF $\approx$ 7.6	2021
5	[10]	Systematic review	Computers and Education: Artificial Intelligence	IF $\approx$ 7.6	2024
6	[2]	arXiv preprint	arXiv (PolicyBot)	N/A	2025
7	[4]	Journal article	Discover Education	New journal	2025
8	[18]	Journal article	Int. J. Advanced Comp. Sci. and Applications	IF $\approx$ 1.5	2024
9	[1]	Conference paper	ACL-IJCNLP	CORE A*/h5 = 207	2021
10	[20]	Journal article	Int. J. Mobile Human Computer Interaction	IF $\approx$ 1.0	2023

Appendix C

Scopus Search Strings Used During the Literature Review

The Scopus search strings below were used to identify the publications reviewed in Chapter 2 and listed in Appendix B. Progressive narrowing was applied: a broad string first, then refinements by venue type, year, and language.

Table C.1: Scopus search strings used during the literature search, listed in the order in which they were issued so that the supervisor can reproduce the exact retrieval chain that produced the ten publications recorded in Appendix B. Each row reports the Boolean string exactly as it was entered into the Scopus interface, the calendar month in which the search was executed, and the number of records the search returned at that step. Progressive narrowing was applied: a broad string was issued first, then refinements were added to filter by venue type, publication year window, and language. The records counts thus give an audit-trail proxy for both the recall of each search and the cumulative reduction in candidate volume from the initial pool to the final selected ten publications.

#	Search string	Date	Records
1	`TITLE-ABS-KEY ( ( "chatbot" OR "conversational agent" ) AND ( "university" OR "higher education" OR "campus" ) )`	2025-11	1,214
2	`TITLE-ABS-KEY ( ( "chatbot" OR "conversational agent" ) AND ( "university" OR "higher education" ) ) AND PUBYEAR > 2018`	2025-11	582
3	`TITLE-ABS-KEY ( ( "retrieval augmented generation" OR "RAG" ) AND ( "policy" OR "institutional" OR "knowledge base" ) )`	2025-11	251
4	`TITLE-ABS-KEY ( ( "intent classification" OR "intent detection" ) AND "chatbot" AND ( "student" OR "university" ) )`	2025-11	132
5	`TITLE-ABS-KEY ( ( "machine translation" OR "low resource" ) AND ( "Luganda" OR "Acholi" OR "Runyankole" OR "Swahili" OR "Bantu" ) )`	2025-11	89
6	`TITLE-ABS-KEY ( ( "hybrid retrieval" OR "lexical" ) AND ( "vector search" OR "dense passage" OR "semantic search" ) )`	2025-11	197
7	`TITLE-ABS-KEY ( "WhatsApp" AND ( "chatbot" OR "bot" ) AND ( "student" OR "education" OR "Africa" ) )`	2025-11	73
8	`TITLE-ABS-KEY ( ( "sentence transformer" OR "MiniLM" OR "SBERT" ) AND ( "embedding" OR "similarity" ) )`	2025-11	303

Keyword priority. The highest-priority keywords were “chatbot”, “university”, and “retrieval augmented generation”. Secondary keywords included “intent classification”, “hybrid retrieval”, “low resource”, “WhatsApp”, and “SBERT”. Filters applied at the Scopus interface included `LANGUAGE (English)`, `DOCTYPE (article OR review OR conference paper)`, and `PUBYEAR > 2018`; foundational works pre-dating 2018 were obtained by citation chasing.

Appendix D

Research Alignment with Specialization, SDG, NDPIV and Vision 2040

Alignment with Programme Specialization

The project aligns with the **Artificial Intelligence and Robotics** specialization track. The retrieval-and-generation pipeline implemented in Chapter 3 draws directly on the natural-language processing material covered in the AI and machine-learning electives: sentence-transformer embeddings, vector similarity search, intent routing, and the integration of a hosted large language model. The vision module extends the same competencies to image understanding through a multimodal model. The design science evaluation framework in Section 3.7 demonstrates the empirical and analytical skills the track is designed to develop.

Alignment with Sustainable Development Goals (SDG)

SDG 4: Quality Education. The system reduces the barriers that students at a sub-Saharan African university face when seeking accurate, up-to-date information about their education.

SDG 9: Industry, Innovation and Infrastructure. The system demonstrates how a small institution in a resource-constrained setting can build production-grade AI infrastructure using only free-tier cloud services.

SDG 10: Reduced Inequalities. The multilingual layer (English plus six Ugandan languages) and the voice input and output features reduce the inequalities that English-only digital services impose on students more comfortable in a Ugandan language or with limited typing literacy.

Alignment with the National Development Plan IV (NDPIV)

The project aligns with NDPIV objectives on human-capital development and ICT-enabled service delivery. By demonstrating that a Ugandan university can deliver a

usable, multilingual, mobile-accessible information service using only freely available cloud infrastructure, the project provides a replicable template that other UCU faculties and Ugandan tertiary institutions can adopt without large capital outlays.

Alignment with the Uganda Vision 2040

Vision 2040 sets out the ambition for Uganda to become a modern, prosperous country with a knowledge-based economy. The present project contributes by (i) building and deploying a working AI artefact in a Ugandan institutional context, (ii) respecting the linguistic diversity of the Ugandan population, and (iii) demonstrating that Ugandan students can produce AI systems combining retrieval, generation, vision, and translation modules to address authentic local problems.

Appendix E

Other Useful Figures

This appendix presents supplementary material: the database schema underlying the analytics dashboard, the configurable system parameters used in the deployed configuration, and an additional WhatsApp screenshot showing grounded responses to student-conduct queries.

Database schema.

Table E.1: Database tables and columns underlying the deployed system's persistence layer. Four logical tables are managed by a single PostgreSQL schema (with SQLite fallback for local development): `users` stores authenticated accounts with Werkzeug-salted password hashes, `chats` records the per-user conversation history that powers the analytics dashboard's per-language metrics, `interaction_events` is the per-request analytics log from which every figure in Chapter 4's quantitative section is derived (success/fallback flag, source language, translation flags, latency, error category), and `data_deletion_requests` is the audit trail that backs the right-to-erasure endpoint declared in Section 3.9. Together these four tables are the data foundation that makes the system's evaluation reproducible and its compliance posture verifiable.

Table	**Column**	**Description**
users	id	Auto-incrementing primary key
	username	Unique login name
	password_hash	Salted hash (Werkzeug)
	role	`student` or `admin`
	created_at	Registration timestamp
chats	id	Auto-incrementing primary key
	user_id	Foreign key to users
	question	User query text
	answer	Generated response text
	created_at	Interaction timestamp
interaction_events	id	Auto-incrementing primary key

Table E.1 – continued

Table	Column	Description
	channel	`web` or `whatsapp`
	user_ref	User identifier
	question_text	Query (max 1 000 chars)
	answer_text	Response (max 4 000 chars)
	source_language	Detected language code
	translated_inbound	1 if inbound translation applied
	translated_outbound	1 if outbound translation applied
	success	1 if successful
	fallback_used	1 if fallback triggered
	latency_ms	End-to-end latency
	error_type	Error category if failed
	created_at	Event timestamp
data_deletion_requests	id	Auto-incrementing primary key
	full_name	Requester name
	contact_email	Requester email
	whatsapp_number	Optional phone
	details	Deletion request details
	status	`pending` by default
	created_at	Request timestamp

Configurable system parameters.

Table E.2: Configurable system parameters with their default values and operational role. The table records every parameter that we treat as a tunable knob during deployment: the embedding model and the language model selected at the cloud layer, the chunk-size and chunk-overlap window that govern how documents are segmented before indexing (justified in Section 3.5), the per-branch top-k values that bound the retrieval cost, the merged-context limit fed to the LLM prompt, the temperature and maximum-token budget on the generator, and the Qdrant collection identifier and distance metric used by the vector store. Together these parameters describe the full configuration surface a supervisor or reviewer would need to reproduce the deployed system; the defaults shown are those used during the bench and production-traffic evaluation reported in Chapter 4.

Parameter	**Default Value**	**Description**
Embedding model	all-MiniLM-L6-v2	Sentence transformer for chunk encoding
LLM model	llama-3.1-8b-instant	Cloud-hosted generation model
Chunk size	300 words	Maximum words per chunk
Chunk overlap	60 words	Overlap between consecutive chunks
Semantic top-k	6	Chunks returned by vector search
Lexical top-n	8	Chunks returned by keyword search
Lexical min hits	1	Minimum query terms in a matching line
Merged limit	4	Chunks sent to the generator
Generation temperature	0.3	LLM sampling temperature
Max tokens	512	Maximum generation length
Collection name	ucu_docs	Qdrant collection
Distance metric	cosine	Vector similarity measure

Appendix F

Research Poster Layout

The one-page research poster produced for this project is reproduced below. It summarises the research concept, the problem addressed, the objectives, the methodology, and the expected results.

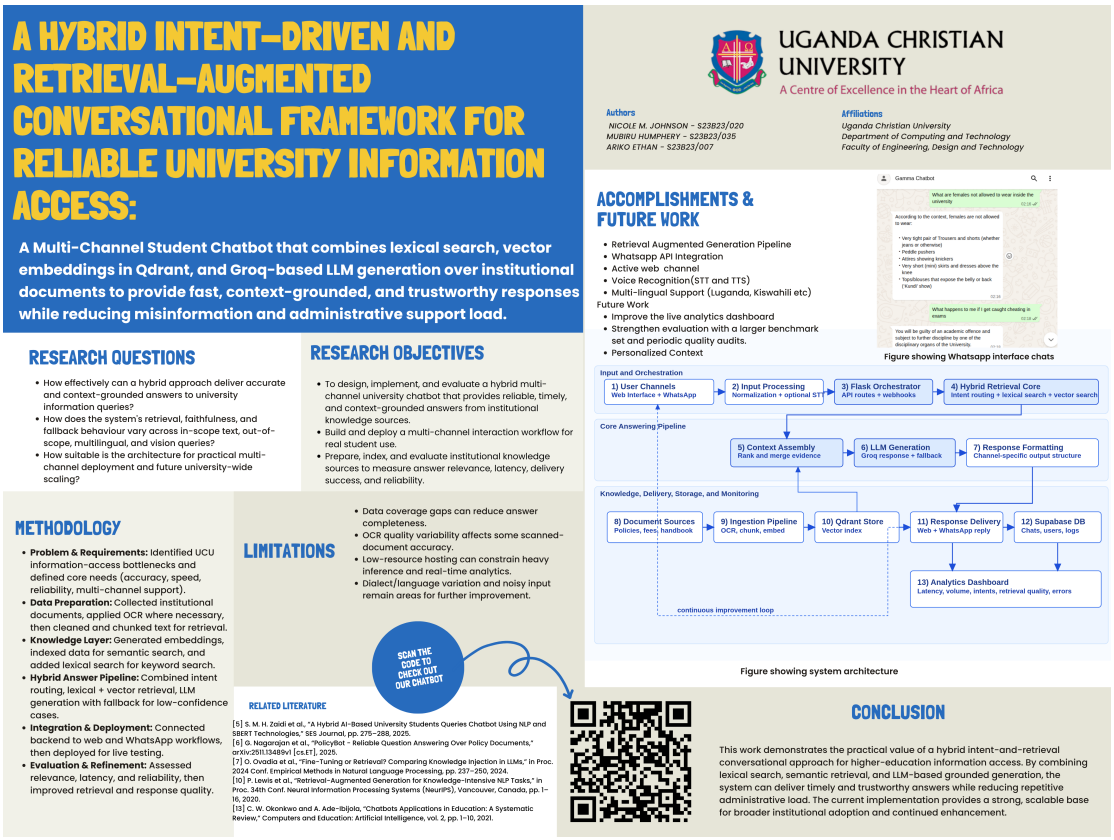


Figure F.1: One-page research poster summarising the Group Gamma chatbot project. The poster condenses the full submission into a single A1-format panel: it states the research concept, the problem addressed (fragmented institutional information delivered through inconsistent channels), the three specific objectives, the high-level methodology (hybrid lexical-plus-vector retrieval, LLM grounding, confidence-gated fallback, multilingual pivot, multi-channel delivery), and the expected results (grounded answers, fallback routing, cross-channel parity)