Report

The University of Missouri–Kansas City

Dr. Yugyung Lee

January 28th, 2026

CS 5588

Team Members:

Lyza Iamrache

Ailing Nan

Gia Huynh

1. Introduction and Problem Definition

Universities maintain large volumes of institutional information across distributed documents such as catalogs, policies, transportation schedules, safety reports, and campus maps. While this information is publicly available, it is often stored in long PDF documents that require manual searching and interpretation. Students, visitors, and staff frequently struggle to locate accurate answers quickly, especially when decisions are time-sensitive (e.g., transportation schedules, parking rules, or academic requirements).

The primary problem addressed in this project is **inefficient access to trusted campus information**. Traditional keyword search or manual browsing introduces delays, inconsistent interpretation, and reduced user confidence.

The objective of SmartCampus is to develop a **trust-aware Generative AI digital twin** capable of:

- Retrieving relevant institutional knowledge from official UMKC documents,
- Producing grounded responses supported by evidence,
- Providing transparent confidence indicators,
- Supporting real-world decision workflows through a deployable application interface.

Unlike generic chatbots, SmartCampus emphasizes **verification and reliability**, ensuring responses are traceable to official university sources.

2. System Architecture

SmartCampus follows a modular architecture designed for scalability, monitoring, and deployment readiness.

Architecture Overview

User → Streamlit Interface → Retrieval Pipeline →

Evidence Selection → Grounded Response →

Logging & Evaluation → Analytics Dashboard

Core Components

1. Data Layer

Institutional documents serve as the knowledge base, including:

- UMKC University Catalog
- Shuttle Schedules
- Parking Permit Documentation
- Clery Safety Reports
- Campus Maps
- Visual Identity Guidelines

These documents represent authoritative university knowledge sources.

2. Processing Layer

PDF documents are ingested and transformed into structured text using automated extraction pipelines. The system performs normalization and segmentation to prepare data for retrieval.

3. Retrieval Layer

A retrieval module identifies relevant document segments using keyword overlap scoring and ranked selection.

4. Generation Layer

Responses are generated using retrieved evidence, ensuring outputs remain grounded in institutional data.

5. Monitoring & Evaluation Layer

All interactions are logged and evaluated using structured metrics stored locally and within an evaluation database.

This architecture separates concerns between data ingestion, reasoning, and monitoring, enabling future extensibility.

3. Data Engineering and Processing

Data Sources

The SmartCampus system integrates multiple institutional PDFs representing diverse operational domains:

Domain	Example Documents
Academic	University Catalog
Transportation	Shuttle Schedule
Parking	Permit Policies
Safety	Clery Report
Navigation	Campus Maps

Branding	Visual Guidelines

These sources ensure responses originate from verified institutional materials.

PDF Processing Pipeline

The ingestion system performs:

1. PDF parsing using PyMuPDF
2. Text normalization
3. Chunk segmentation
4. Metadata tagging

Chunking Strategy

Documents are split into overlapping text segments:

- Chunk size: **1200 characters**
- Overlap: **200 characters**

Chunking improves retrieval accuracy by preserving context while enabling efficient search.

Each segment receives a unique identifier:

evidence_id = document#page#chunk

This identifier enables traceable citations and logging.

4. Retrieval and Model Design

SmartCampus implements a Retrieval-Augmented Generation (RAG) workflow.

Retrieval Method

The retrieval engine:

1. Accepts user query text
2. Computes keyword overlap scores
3. Ranks document chunks
4. Returns Top-K evidence segments

This baseline approach ensures deterministic and explainable behavior.

Grounded Response Generation

Responses are produced using retrieved evidence rather than external knowledge sources. Each answer references supporting evidence IDs to improve transparency.

Example citation:

[umkc_catalog#p45#c2]

Trust Mechanisms

To reduce hallucination risk, the system introduces:

- **Confidence Score:** derived from retrieval strength

- **Faithfulness Indicator:** evaluates reliability

- **Refusal Condition:** triggered when confidence is low

If confidence < threshold → system refuses to answer

This design prioritizes correctness over completeness.

5. Evaluation and Monitoring

Evaluation is treated as a product-level capability rather than an offline experiment.

Logged Metrics

Each query records:

- latency (ms)
- rows returned
- average retrieval score
- keyword count
- version identifier
- timestamp

Metrics are stored in an evaluation table (EVAL_METRICS) within Snowflake.

Impact Metrics

The project evaluates user impact using:

- **Time-to-decision reduction**

  (baseline manual search ≈ 7 minutes)

- **Trust verification rate**

  percentage of responses with citations

- **Adoption signal**

  number of queries executed

Technical Metrics

System-level performance includes:

- citation coverage
- mean retrieval score
- latency p50 and p95
- confidence distribution
- version comparison analytics

Evaluation dashboards enable comparison across system versions.

6. Deployment and Application Interface

A Streamlit application exposes the SmartCampus system as an interactive product interface.

Application Features

The interface provides:

- user query input

- grounded response display
- evidence preview panel
- metrics visualization
- automatic logging

Users can immediately verify information sources supporting each answer.

Deployment Readiness

The system includes:

- modular repository structure
- reproducible requirements.txt
- automated ingestion scripts
- evaluation persistence layer
- environment-based configuration

These components support rapid deployment to Streamlit Cloud or similar platforms.

7. Failure Analysis and Risk

Failure Scenario

Retrieval selects an incorrect document section, producing a misleading grounded answer.

Example:

A student requests shuttle hours during finals week, but retrieval returns a general transportation policy instead of the updated schedule.

Impact

Potential consequences include:

- missed transportation
- safety risks during late hours
- reduced user trust

Detection Signals

Failures are detected via:

- low confidence scores
- missing citations
- increased repeated queries
- abnormal latency patterns

Mitigation Strategies

- refusal when confidence is low
- improved chunk ranking
- domain-aware retrieval filtering
- continuous evaluation monitoring

8. Conclusion and Future Work

Phase-2 demonstrates the feasibility of a trust-aware campus digital twin capable of transforming institutional documents into actionable knowledge.

Key achievements include:

- automated document ingestion pipeline
- retrieval-augmented reasoning workflow
- evidence-grounded responses
- deployment-ready application interface
- evaluation and monitoring framework

Future development will focus on:

- hybrid semantic retrieval
- multimodal campus map reasoning
- domain classification before retrieval
- real-time institutional data integration
- expanded evaluation automation

SmartCampus establishes a foundation for trustworthy AI-assisted decision support within university environments.