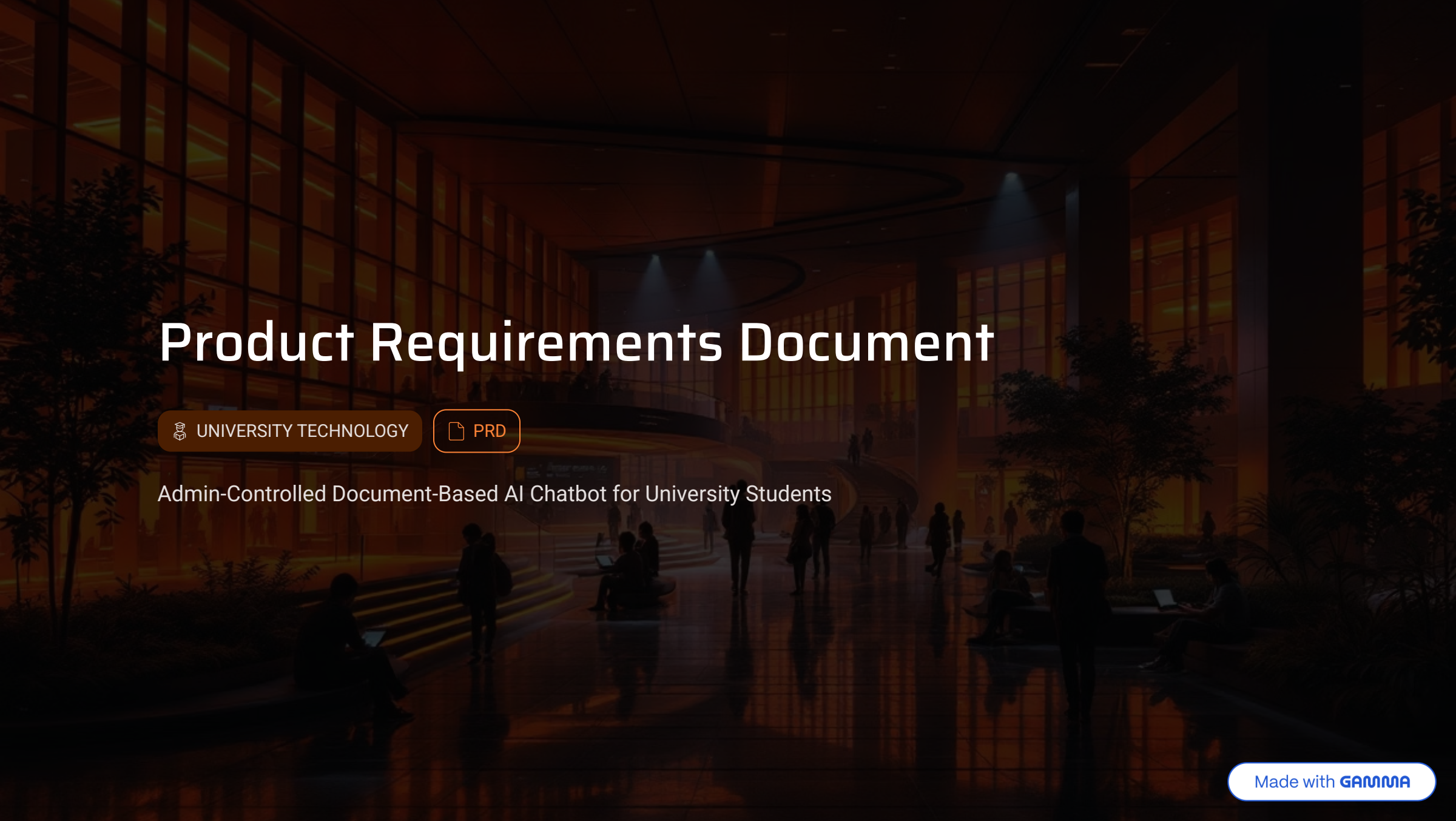


Product Requirements Document

UNIVERSITY TECHNOLOGY

PRD

Admin-Controlled Document-Based AI Chatbot for University Students

Introduction

Universities generate a large volume of academic documents such as syllabi, lecture notes, regulations, timetables, and internal guidelines. Students often rely on informal sources or general-purpose AI chatbots, which may provide incorrect or unofficial information.

This project aims to develop an **AI-powered chatbot that answers student queries strictly based on administrator-uploaded academic documents**, ensuring accuracy, reliability, and institutional control.

The system follows a privacy-aware, cost-free, and open-source-first approach, making it suitable for academic deployment and evaluation.

Problem Statement

Hallucinated Answers

AI tools generate plausible-sounding but factually incorrect responses

External Knowledge

Chatbots use internet sources instead of official university documents

No Institutional Control

Universities cannot verify or manage the information students receive

Students lack a centralized, reliable system to query official university documents. These challenges lead to misinformation, confusion, and reduced trust in academic guidance systems. The absence of document-based verification means students may receive conflicting information that doesn't align with current institutional policies or course requirements.

Proposed Solution

Document Upload

Admins control all source content

RAG Processing

Retrieval Augmented Generation ensures accuracy

Controlled Responses

Answers only from verified documents

Student Access

Responsive interface for all devices

Key Innovation: The system does **not store AI models or vector embeddings persistently**, ensuring ethical AI usage and data minimization principles are upheld throughout operation.

User Roles & Permissions

Administrator

- Secure login authentication
- Upload academic documents (PDF, DOCX, PPT)
- Trigger document re-indexing
- Manage document availability and permissions

Student

- Secure login access
- Ask academic questions through chat interface
- Receive answers based exclusively on uploaded documents
- Access chat history and previous interactions

Functional Requirements

Student Interface Module

- Responsive chat interface (mobile, tablet, desktop)
- Question submission via text input
- Display of AI-generated answers
- Fallback message if answer is unavailable

Admin Management Module

- Secure authentication system
- Document upload and management interface
- Supported formats: PDF, DOCX, PPT
- Trigger document re-indexing controls

Document Processing Module

- Text extraction from uploaded documents
- Intelligent chunking for efficient retrieval
- Metadata association with documents

AI & Vector Search Module

Question embedding using hosted API, similarity search using FAISS (in-memory), and retrieval of relevant document chunks for context-aware responses.

Answer Generation & Control Module

Context-restricted answer generation using hosted LLM, prevention of responses outside document scope, and standard fallback response for unavailable information.

Non-Functional Requirements

No Local AI Storage

The system does not store AI models locally, ensuring minimal infrastructure requirements and simplified deployment across institutional environments.

No Persistent Vectors

Vector embeddings exist only in memory during runtime, automatically destroyed on restart to ensure data minimization and privacy compliance.

High Availability

System maintains consistent performance during operational hours with automatic recovery and failover mechanisms for uninterrupted service.

Privacy-Preserving Design

Architecture prioritizes student data protection with minimal data retention and compliance with academic privacy standards.

Open-Source Technologies

Built exclusively with free and open-source tools to ensure transparency, community support, and zero licensing costs for institutions.

Simple Cloud Deployment

Streamlined deployment process compatible with standard cloud platforms, enabling rapid setup and easy maintenance.

System Architecture Overview

Retrieval Augmented Generation (RAG) Pattern

01

User Question Submission

Student enters query through responsive chat interface

02

Embedding Conversion

Question converted to numerical vector representation

03

Similarity Search

FAISS performs in-memory vector comparison

04

Document Retrieval

Most relevant document chunks are identified and extracted

05

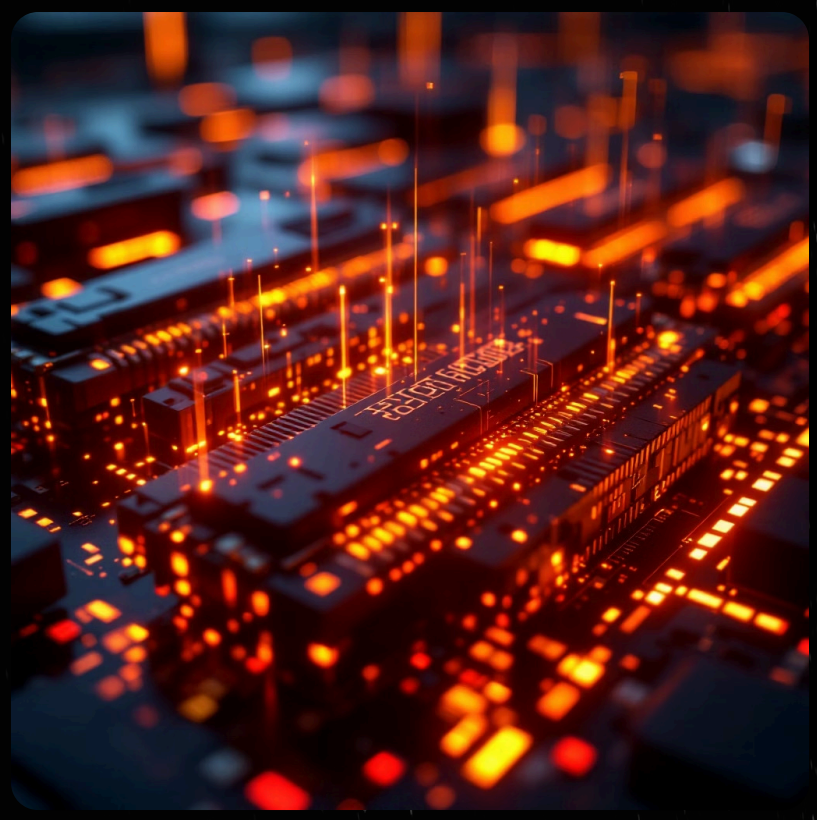
Answer Generation

Hosted LLM generates context-restricted, accurate response

This architecture ensures all answers are grounded in official university documents, eliminating hallucinations and maintaining institutional control over information accuracy.

Vector Search Design

KEY REQUIREMENT



In-Memory Architecture

The FAISS index exists **exclusively in memory (RAM)** during application runtime. This fundamental design choice has profound implications for data privacy and system behavior.

Application Startup

Index rebuilt automatically from stored documents

1

Runtime Operation

Embeddings exist only in volatile memory

2

Container Restart

All embeddings destroyed completely

3

Zero Long-Term AI Data Retention

No persistent storage of vector embeddings or model artifacts

Data Minimization Compliance

Adherence to privacy principles and data protection regulations

Academic & Ethical AI Usage

Responsible AI implementation suitable for educational institutions

No .index or embedding files are saved to disk. This ensures complete data ephemerality and prevents unintended long-term storage of AI-processed information.

Technology Stack

Frontend

- HTML5
- Vanilla JavaScript
- Tailwind CSS (CDN)
- Fully responsive UI

Backend

- Python 3
- Flask framework

AI Services (Hosted)

- Hugging Face Inference API (LLM)
- Hugging Face Embedding API

Vector Search

- FAISS (in-memory only)

Database & Storage

- Supabase PostgreSQL (users, roles, metadata)
- Supabase Storage (uploaded documents)

This carefully selected technology stack prioritizes open-source solutions, cost efficiency, and academic suitability while maintaining enterprise-grade reliability and performance. Each component has been chosen to support the system's core principles of privacy preservation, institutional control, and ethical AI deployment.