EFFICIENT COURSE INFORMATION RETRIEVAL: AI-POWERED UNIVERSITY SYLLABUS CHATBOT

Final Project Report

By

20BCE0024 - Tharun Jayaprasad

20BCE0031 - Chitteshwari Satish

20BCE0569 - Sanjna Srivastava

20BCE0106 - MD Danish Anwar

in partial fulfilment for

HUMAN COMPUTER INTERACTION (CSE4015)

Under the Guidance of

Dr. Swarnalatha. P

School of Computer Science and Engineering

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

JULY 2023 - DECEMBER 2023

INDEX

1. Introduction

1.1. Scope

The scope of this endeavour is to create an all-encompassing open-source chatbot tailored to serve the Computer Science department at VIT comprehensively. It will enable students and faculty to access detailed information about each core course, including the breakdown of lab work, theory syllabus, and project components. The chatbot will provide insights into the credit distribution for every course, facilitating a deeper understanding of the academic structure. Moreover, it will assist users in identifying and fulfilling course prerequisites, ensuring a smooth academic journey, without getting hassled by the cryptic syllabi found on VTOP.

1.2. Objective

The main objective of this paper is to introduce a versatile and comprehensive chatbot to the Computer Science department at VIT. This chatbot will serve as a centralised resource for students and faculty seeking information about core courses, offering insights into both theoretical and practical components, including lab assignments and project work. Additionally, it will provide clarity on course credits, helping students make informed academic decisions.

Another important aim of the chatbot is to simplify the process of identifying and fulfilling prerequisites for each course. It will offer guidance on the foundational knowledge needed for success in specific courses, ultimately enhancing students' academic achievements and satisfaction. The open-source nature of the chatbot underscores its adaptability and sustainability, ensuring it can evolve to meet the changing needs of the Computer Science department at VIT, ultimately leading to improved academic outcomes and enhanced student experiences. Presentations are communication tools that can be used as demonstrations, lectures, speeches, reports, and more.

1.3. Abstract

The VIT Syllabus Helper is a LLM chatbot designed for core courses in the Computer Science department at Vellore Institute of Technology (VIT). The chatbot leverages a combination of state-of-the-art open-source technologies to provide robust functionality, including Sentence Transformers for

embeddings, FAISS CPU for vector storage, and integration of Llama 2, a large language model, using the Chainlit library for an interactive conversational interface. The project's key focus is on improving the learning experience for students at VIT by providing an intelligent and efficient tool for accessing course-related information. By utilising Sentence Transformers, the chatbot can understand and process natural language queries with higher accuracy and context awareness. Additionally, Faiss CPU is employed to efficiently store and retrieve vector representations of text, compensating for GPU's speed with its easy accessibility.

1.4. Requirement Specification

1.4.1. Functional and Non-Functional Requirements

Functional Requirements:

- Course Specific Information: Code, Objective and Outcome, Credits, Pre-requisites, Module-wise breakdown, prescribed reference materials.
- Natural Language description: The information should be provided in a sentence form, so that even novice users can comprehend.
- In case the bot is unaware of the information, the bot must not provide false, non-existent information.

Non-functional Requirements:

- Scalability: Accommodate future course additions.
- Privacy: Comply with privacy regulations and ensure the confidentiality of user data.
- Feedback: Collect user suggestions and comments for continuous improvement.

1.4.2. Software Requirements

The software requirements for the successful implementation of the proposal are:

- Windows OS
- Python 3.x
- Libraries: Langchain, Chainlit
- Git

- Any text editor or IDE.

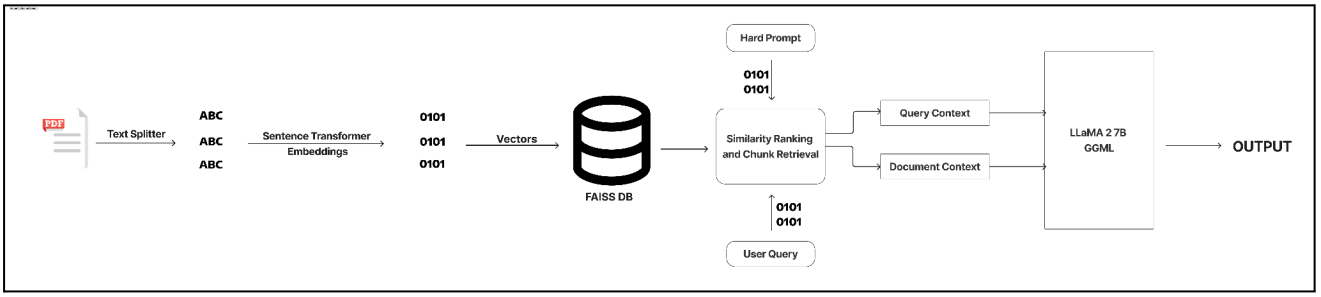
1.4.3. Hardware Requirements

The hardware requirements for the successful implementation of the proposal are:

- CPU
- 64-bit RAM
- ~10 GB system Memory
- Reliable internet connectivity

1.5. Process Model of the System

The process model of the system describes the high-level functional working of the chatbot system. The custom data in PDF format is split into smaller chunks of fixed size using a Character Text Splitter. These chunks are then converted into vectors or embeddings using Sentence Transformers that have Python bindings. These vectors can then be stored in the local machine using FAISS Vector Database for quick retrieval. A hard prompt coded into the program assists the Large Language Model - Llama 2 to process the user input known as prompt. Using the embeddings extracted from FAISS using its similarity searching feature, Llama 2 gives an appropriate response to the user query.



Process Model

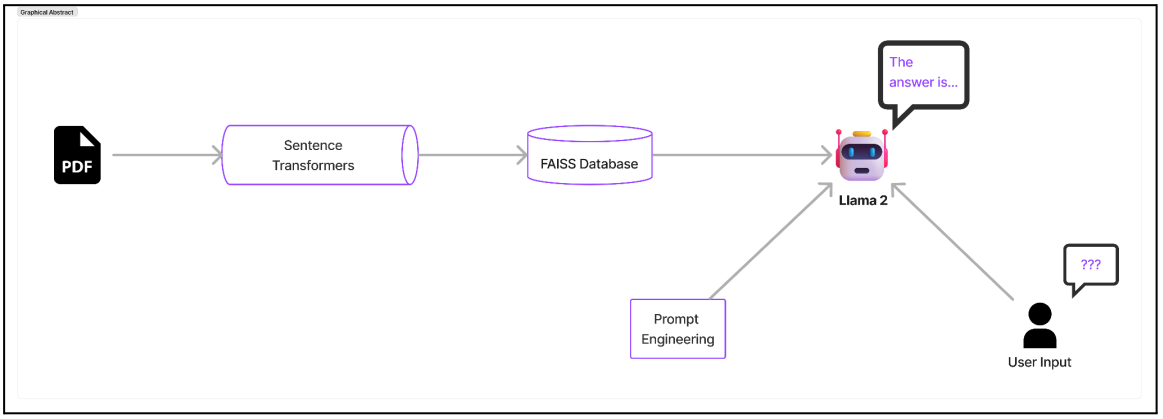
1.6. System Architecture

The system architecture is a diagram of the system that shows its foundational structure and the layout of the different components of the system that work together. Here, the different components are the

- PDF Data
- Sentence Transformers

- FAISS Database
- LLama 2 (7B parameters quantized model)
- Hard Prompt
- User Input

The block diagram shows the interaction between these elements of the system which helps in creating a strong basis for the design and implementation.



Architecture of the System

2. Literature Survey

In their research paper, 'The emergent role of artificial intelligence, natural learning processing, and large language models in higher education and research' the authors, Tariq Alqahtani, Hisham A. Badreldin, Mohammed Alrashed, Abdulrahman I. Alshaya and Sahar S. Alghamdi thoroughly examines the transformative impact of Artificial Intelligence (AI), Natural Language Processing (NLP), and Large Language Models (LLMs) in higher education and research. The exploration encompasses applications such as personalised learning, grading, curriculum design, and career guidance in education, delving into AI's contributions to text generation, data analysis, literature reviews, and peer review processes in research. The survey also addresses the emerging use of NLP models for mental health support. Throughout the paper, the author emphasises the imperative for responsible AI use, ethical considerations, and advocates for a balanced integration of AI and human support to optimise outcomes in education and research.

The literature survey of the paper,' Leveraging Large Language Models to Power Chatbots for Collecting User Self-Reported Data', explores the application of Large Language Models (LLMs), such as GPT-3, in the development of chatbots for

collecting user self-reported data in the context of digital health. The authors, Jing Wei, Sungdong Kim, Hyunhoon Jung, and Young-Ho Kim, highlight the limitations of existing commercial chatbot frameworks, emphasising the need for more flexible and dynamic conversational agents. The study investigates the potential of LLMs, with billions of pre-trained parameters, to power chatbots capable of engaging in naturalistic conversations and effectively collecting self-report data on health-related topics. The research evaluates the impact of prompt design factors, including information specification format and personality modifiers, on the slot-filling ability and conversation styles of the resulting chatbots. The authors discuss the advantages, such as versatile responses, context tracking, and low-effort bootstrapping, as well as drawbacks, including randomness and repetitiveness, of LLM-driven chatbots. The study also addresses ethical considerations and proposes strategies to mitigate potential issues. Overall, the literature survey contributes empirical insights into the feasibility and challenges of leveraging LLMs for developing chatbots focused on data collection in the field of personal informatics.

The literature survey of the paper,' Llama 2: Early Adopters' Utilisation of Meta's New Open-Source Pretrained Model ', delves into the burgeoning field of artificial intelligence (AI), focusing on the introduction and early adoption of Llama 2, an open-source pre-trained model released by Meta. Authored by Konstantinos I. Roumeliotis, Nikolaos D. Tselikas, and Dimitrios K. Nasiopoulos, the survey investigates the foundational elements of Llama 2 and explores how early adopters leverage its capabilities in AI projects. The authors emphasise the significance of understanding the perspectives and experiences of these early adopters, who play a pivotal role in driving innovation and providing insights for further model enhancements. The survey provides insights into the strengths, weaknesses, and areas of improvement of Llama 2, offering valuable guidance for the AI community and Meta to enhance future model iterations. Furthermore, it discusses the implications of Llama 2's adoption on the broader open-source AI landscape, addressing challenges and opportunities for developers and researchers. This early exploration of the Llama 2 pre-trained model serves as a foundational basis for future research investigations, shedding light on the practical implications and ethical considerations associated with its use in the evolving landscape of AI technologies.

The paper titled "Chat Vector: A Simple Approach to Equip LLMs with New Language Chat Capabilities" by Shih-Cheng Huang et al. explores the development of Large Language Models (LLMs) for non-English languages, with a focus on aligning them with human preferences. The authors propose a computationally efficient method called "chat vector," which involves restructuring the conventional training paradigm from continual pre-training, Supervised Fine Tuning (SFT), and Reinforcement Learning from Human Feedback (RLHF) to continual pre-train + chat. The chat vector is derived by subtracting the pre-trained weights of a base model (LLaMA2) from its chat-enhanced counterpart (LLaMA2-chat). The empirical studies primarily focus on Traditional Chinese, with evaluations based on toxicity, ability to follow instructions, and multi-turn dialogue. The results demonstrate the chat vector's efficacy in improving conversational skills. The approach is extended to models pre-trained in Korean and Simplified Chinese, showcasing its versatility. The paper contributes a significant solution for aligning LLMs with human preferences efficiently across various languages, achieved through the innovative concept of the chat vector.

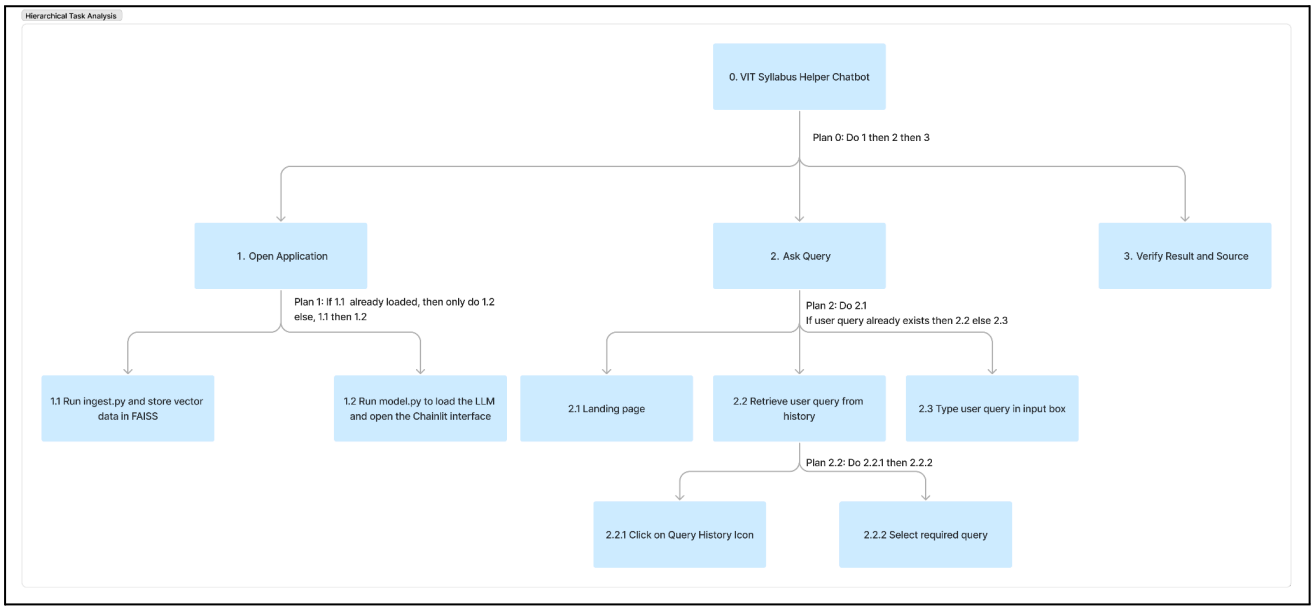
The literature survey of the paper, ‘Effects of Generative Chatbots in Higher Education’, reveals a growing interest in the transformative impact of generative chatbots in higher education. Ilieva et al. (2023) highlight the challenges faced by traditional learning technologies in providing interactive and real-time feedback while emphasising the potential of intelligent chatbots based on generative artificial intelligence (AI) to address these shortcomings. The authors propose a theoretical framework for blended learning with chatbot integration, offering advantages such as comprehensive understanding, enhanced educational experiences, and unified applications in teaching–learning activities within universities. The study underscores the role of generative chatbots in guiding both students and instructors, streamlining pedagogical activities, and reducing the workload on educators. Additionally, the paper explores the characteristics of existing educational chatbots, emphasizing their ability to provide conversational assistance, support multi-modality, offer multilingual capabilities, and integrate with other software systems. The survey also touches upon the rapid growth of the conversational AI market, with generative chatbots anticipated to play a significant role in reshaping the educational landscape. Furthermore, the study by Ilieva et al. (2023) provides insights into the application of chatbots in higher education, focusing on the assessment of their influence on students' learning

experiences and instructors' teaching methods. The proposed conceptual framework contributes to the systematic evaluation of perceptions and readiness for chatbot-based learning, emphasising the need for institutions to develop AI adoption strategies and invest in digital innovations. The literature survey, therefore, demonstrates a consensus on the potential of generative chatbots to revolutionise teaching and learning practices in higher education.

3. Design Specification

3.1. Hierarchical Task Analysis

Hierarchical Task Analysis (HTA) is a structured method used to analyse a complex system by breaking it down into smaller, manageable tasks. In the context of sign language recognition, HTA can be used to analyse the different steps involved in recognizing sign language gestures. The HTA for the Course Information Question Answering Chatbot can be described as follows:



HTA for the Chatbot

0. VIT Syllabus Helper Chatbot
1. Open Application
    a. Run ingest.py and store vector data in FAISS
    b. Run model.py to load the LLM and open the Chainlit interface
2. Ask Query
    a. Landing Page

b. Retrieve user query from history
    i. Click on Query History Icon
    ii. Select required query
c. Type user query

3. Verify result and source

By analysing each broken down step in the process, it is possible to identify potential problems and opportunities for optimization, leading to a more accurate and efficient course information chatbot system.

3.2. Interaction Design

3.2.1. Stakeholder Identification

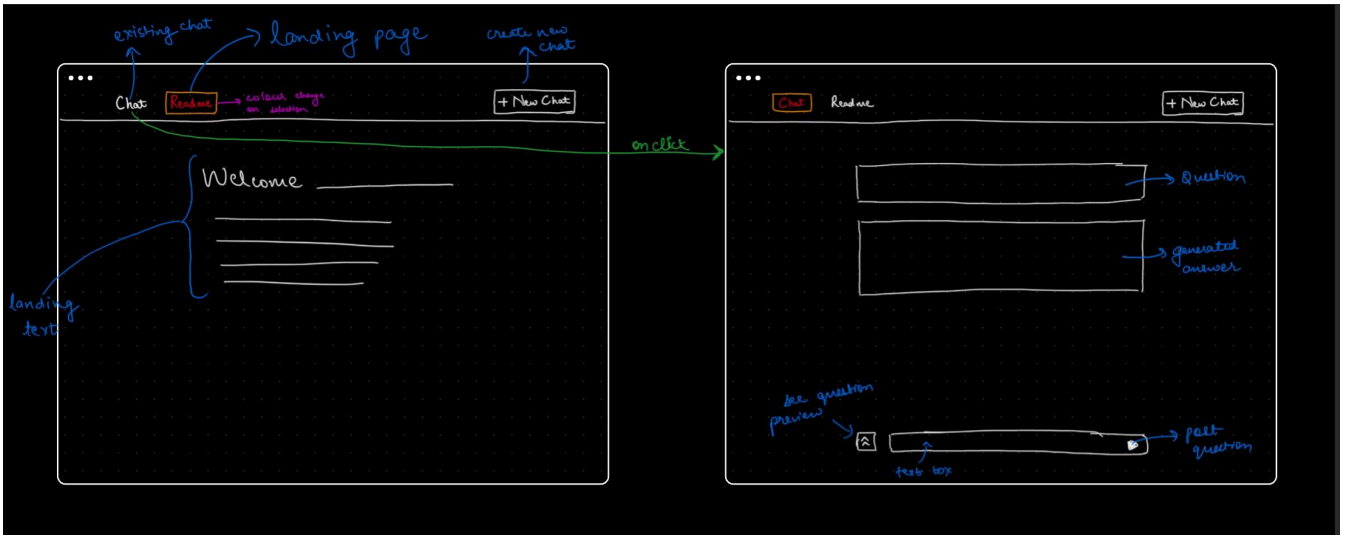
The stakeholder identification was done to explicitly highlight all the stakeholders in the project, whether primary, secondary, direct or indirect. This helps in creating a more user-centric system and tailoring the application to suit the needs of each stakeholder. It also ensures data security. The various stakeholders identified are:

- Faculty Members: Faculty members are essential stakeholders as they will rely on the chatbot to provide accurate and up-to-date course information.
- Students: Students are the primary end-users of the chatbot. They will use it to access information about core courses, lab assignments, syllabi, and project guidelines.
- Administrators: System administrators responsible for maintaining and managing the chatbot's operation, security, and access control are critical stakeholders. They ensure the chatbot remains functional and secure.
- Course Coordinators: Individuals responsible for coordinating and managing specific courses will play a role in ensuring the chatbot has accurate and timely information related to each course.

3.2.2. Storyboarding

The storyboard allowed us to visualise and refine the user experience, interaction flow, and conversational journey within the context of a chatbot application. Its key elements include:

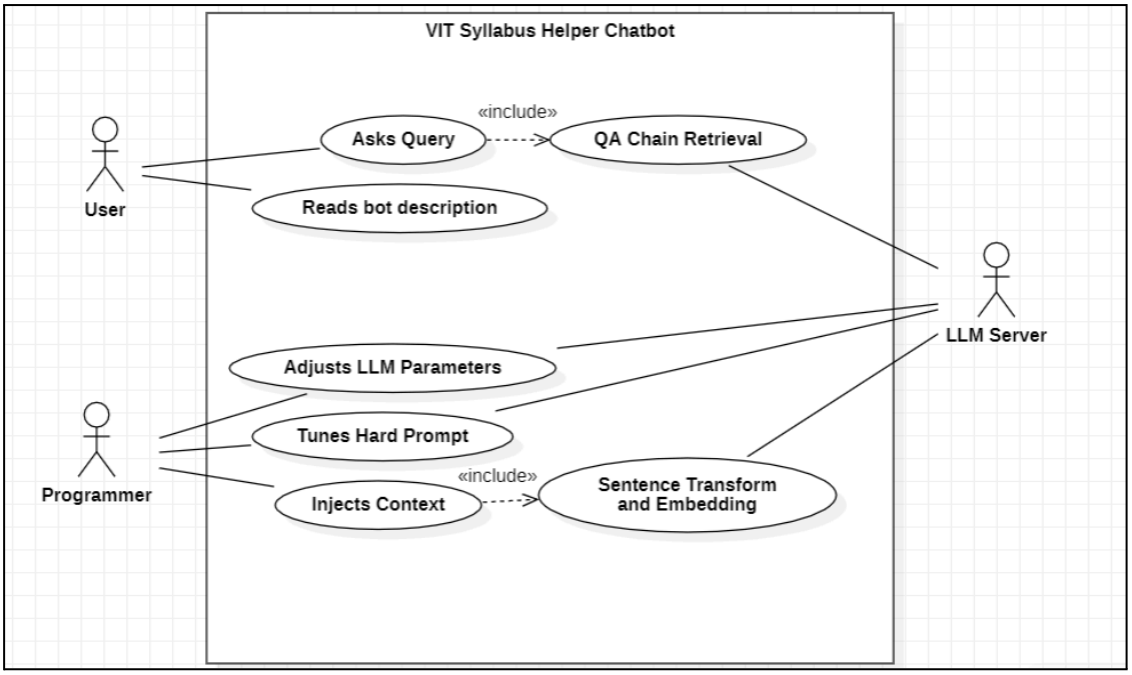
- User queries
- Bot responses
- User flow and navigation
- Error handling



Storyboard Visualization of the Chatbot

3.2.3. Use Case Modelling

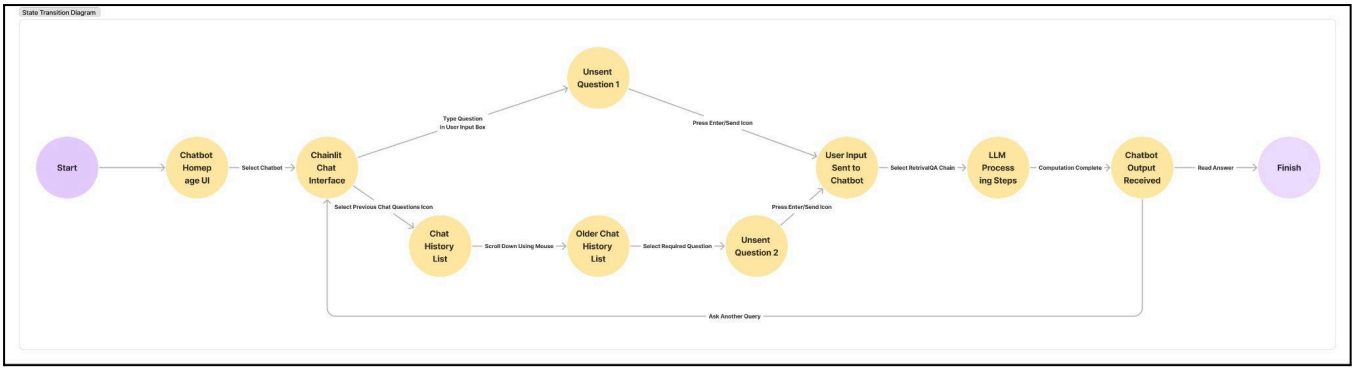
The actors identified for the VIT Syllabus Chatbot were the end users (students, faculties, researchers, etc.) querying the chatbot, the programmers (i.e, us) who are responsible for the development and maintenance of the bot and finally the LLM Server, which in our case is the quantised LLaMA 2.0 7B model, hosted locally on the CPU machine. The Server is responsible for performing the QARetrieval using RAG (Retrieval Augmented Generation) from the large data that the model is trained on, as well as adjusting it to the specific provided corpus of the VIT course syllabi. The usecases depict the nature of interaction between each of these actors.



Use Case Modelling for the Chatbot

3.2.4. State Transition Network

The state transition network (STN) represents the flow of the system from one state to another with the help of associated dialogue in the form of circles and arcs connecting them. Here, in the chatbot system, on startup, the initial Chainlit's UI is visible which in turn leads to the chat interface. Here, users can proceed to the next state by entering a prompt either by manual typing or using the chat history available. The state then transitions to the final state after displaying an output from the Llama 2 LLM for that user query. This is an iterative step since the user can repeatedly use the chatbot to ask any number of questions. This diagrammatic representation of the flow helps in better and more fluid development of the interface and its integration with the system in the backend.



STN for the Chatbot

4. Application of Guidelines

4.1. Application of Schneiderman’s Rules

Shneiderman's eight golden rules are intended to help designers solve problems, and for this purpose Shneiderman offers them significant help with his eight heuristics. In order to improve usability, an interface needs to be well designed to be "user-friendly". The following tabulation is an analysis of how the rules have been applied or obeyed in the proposed and implemented system.

Schneiderman’s Golden Rule	**Implementation in Chatbot**
Strive for Consistency:	All the buttons on the interface are built in a uniform manner with the same style to aid consistency. The inputs from the user and replies from the chatbot are also styled in a similar manner and font to maintain uniformity and aesthetic beauty of the interface.
Enable frequent users to use shortcuts:	Frequent users can make use of the "Previous Chat Questions" option to save time and energy if the question to be asked has already been asked by that user before. Instead of typing the same question again, the frequent/familiar user can scroll through the question history and select the required question directly, with no typing needed. This chat question history is also organised by date and ordered from newest to oldest.
Informative feedback:	The "RetrievalQA" button available on the interface after the user sends an input to the chatbot shows the status of the input processing and output computation in the LLM. Therefore, the user is not kept in the dark and is constantly aware of what is happening in the backend
Design dialog to yield closure:	The chatbot gives the source documents from which the specific answer to a user query has been extracted from which gives closure and confidence to the user about the bot's reply. The chatbot displays a message saying the server cannot be reached

	if the server is overloaded at any moment. This gives the user a better idea of what is going on at the backend, instead of wondering why the computation is taking so long.
Offer Simple Error Handling:	If the user asks questions not relevant to a course syllabus, the chatbot handles that error by saying it does not know the answer.
Permit easy reversal of actions:	"Stop task" allows the user to stop the computation at any stage.
Support internal locus of control:	The chatbot has settings that allow the user to change aspects of the user interface like the theme of the chat (light or dark mode). This gives the user more control over the interface of the bot.
Reduce Short Term Memory Load:	The chatbot is persistent and shows the chat history of a user for a particular active session. As long as the current session is active, the user can scroll back and look at the previously asked and answered queries, thereby reducing the load on their short term memory by not having to remember the conversation with the chatbot. The "Previous Chat Questions" option also saves memory load by storing all the questions asked by the user, including old and new sessions/chats. This chat question history is also organised by date and ordered from newest to oldest. Therefore, the user need not remember all the questions asked in previous sessions of the chatbot.

4.2. Heuristic Evaluation

4.2.1. GOMS

Goals, operators, methods, and selection rules is a method derived from human-computer interaction (HCI) and constructs a description of human performance. Here, two different GOMS analysis are done, one for the process/goal of getting syllabus information and the other for entering a question prompt into the chatbot.

- <u>GOMS for Getting Syllabus Information:</u>

GOAL: READ-VIT-COURSE-SYLLABUS-INFORMATION
[select* GOAL: USE-CHATBOT-METHOD
MOVE-MOUSE-TO-CHATBOT-INPUT-BOX
CLICK-INPUT-BOX
GOAL: USE-TYPE-QUESTION
HOME-TO-KEYBOARD
FORMULATE-QUESTION
TYPE-QUESTION
VERIFY-QUESTION
PRESS-ENTER-KEY

WAIT-FOR-CHATBOT-RESPONSE
READ-PRECISE-CHATBOT-RESPONSE
GOAL: USE-VTOP-METHOD
OPEN-VTOP-WEBSITE
CLICK-ON-STUDENT-LOGIN
GOAL: ENTER-USERNAME
CLICK-ON-USERNAME-INPUT-FIELD
HOME-TO-KEYBOARD
TYPE-USERNAME
VERIFY-USERNAME
GOAL: ENTER-PASSWORD
CLICK-ON-PASSWORD-INPUT-FIELD
HOME-TO-KEYBOARD
TYPE-PASSWORD
VERIFY-PASSWORD
GOAL: ENTER-CAPTCHA
READ-CAPTCHA
CLICK-ON-CAPTCHA-INPUT-FIELD
HOME-TO-KEYBOARD
TYPE-CAPTCHA
VERIFY-CAPTCHA
CLICK-SUBMIT-BUTTON
CLICK-ON-MENU
CLICK-ON-ACADEMICS-SUB-MENU
CLICK-ON-CURRICULUM
CLICK-ON-PROGRAM-CORES-TAB
SCROLL-TO-FIND-COURSE
DOWNLOAD-COURSE-SYLLABUS
OPEN-DOWNLOADED-SYLLABUS
READ-ENTIRE-SYLLABUS]

SELECTION RULES:

RULE 1: If user has chatbot link, use the USE-CHATBOT-METHOD
RULE 2: Else, use the USE-VTOP-METHOD

- <u>GOMS For Chatbot Question Entry:</u>

GOAL: USE-CHATBOT
[select* GOAL: USE-TYPE-METHOD
MOVE-MOUSE-TO-CHATBOT-INPUT-BOX
CLICK-INPUT-BOX
GOAL: USE-TYPE-QUESTION

HOME-TO-KEYBOARD
FORMULATE-QUESTION
TYPE-QUESTION
VERIFY-QUESTION
PRESS-ENTER-KEY
GOAL: USE-QUESTION-HISTORY-METHOD
MOVE-MOUSE-TO-PREVIOUS-CHAT-QUESTIONS-ICON
CLICK-ON-ICON
SCROLL-THROUGH-CHAT-HISTORY-QUESTIONS
CLICK-ON-REQUIRED-QUESTION
VERIFY-QUESTION
PRESS-ENTER-KEY]

SELECTION RULES:
RULE 1: If the question to be asked to the chatbot has already been asked before and is present in the question history, use the USE-QUESTION-HISTORY-METHOD
RULE 2: Else, use the USE-TYPE-METHOD

4.2.2. KLM (Keystroke Level Model)

KLM is done for 3 methods here and their final time values are compared.

a. Native Method via VTOP

Operation	Task description	Time
M	Mental Preparation	1.35
P	Point to Student icon	1.1
B	Click Login Option	0.2
M	Mental Preparation	1.35
P	Point to the Username field	1.1
B	Select the Textbox	0.2
H	Select the Textbox	0.4
M	Mental Preparation	0.35
K*n1	Type Username	0.28 * n1
M	Mental Preparation	1.35

K	Press Tab Key	0.28
M	Mental Preparation	1.35
K*n2	Type Password	0.28 * n2
M	Mental Preparation	1.35
K	Press Enter	0.28
H	Hover to Mouse	0.4
M	Mental Preparation	1.35
P	Point to Academics Menu	1.1
B	Click Academics Menu	0.2
M	Mental Preparation	1.35
P	Point to My Curriculum Submenu	1.1
B	Click My Curriculum Submenu	0.2
M	Mental Preparation	1.35
P	Point to desired subject	1.1
B	Click Download ZIP	0.2
M	Mental Preparation	1.35

Time = (2+n1+n2)*K + 5*P + 5*B + 10*M + 2*H = 21.36 + (n1+n2)*0.28 s

The formula (2+n1+n2)K + 5P + 5B + 10M + 2*H calculates the estimated time for the entire process, emphasising the higher weights on keyboard actions and mouse movements. This method involves moving the mouse, clicking on fields, pressing keys, and hovering over elements, with (n1) and (n2) representing variables for username and password.

b. Using Chatbot Without History

Operation	**Task description**	**Time**
M	Mental Preparation	1.35

P	Point to Chatbot App	1.1
B	Click App	0.2
M	Mental preparation	1.35
P	Point to Input field	1.1
B	Select the Textbox	0.2
H	Hover to Keyboard	0.4
M	Mental Preparation	1.35
K*M1	Type Query	0.28*M1
M	Mental Preparation	1.35
K	Press Enter	0.28
R	System Response	180

Time = (1+m1)*K + 2*P + 2*B + 4*M + 1*H + 1*R = 188.68 + m1*0.28s

The formula (1+m1)K + 2P + 2B + 4M + 1H + 1R calculates the estimated time for the entire interaction, with m1 keystrokes being utilised to type the user query. This method outlines the user's steps, from initiating the chatbot to receiving a response, providing insights into efficiency and guiding design optimizations for chatbot applications.

c. Using Chatbot With History

Operation	**Task description**	**Time**
M	Mental Preparation	1.35
P	Point to Chatbot App	1.1
B	Click App	0.2
M	Mental Preparation	1.35
P	Point to History Icon	1.1
B	Click Icon	0.2
M	Mental Preparation	1.35
P	Point to Desired Query	1.1

B	Select Query	0.2
M	Mental Preparation	1.35
P	Point to Send Query Button	1.1
B	Click Submit	0.2
R	System Response	180

Time = 4*P + 4*B + 4*M + 1*R = 190.6s

The formula 4*P + 4*B + 4*M + 1*R calculates the estimated time for the entire interaction. In practical terms, users navigate to the chatbot, access the history, select a question, send a query, and receive a response. This analysis offers insights into the efficiency of incorporating chat history within the chatbot application, potentially guiding enhancements for improved user experience and system optimization.

5. CCG

5.1. Communication

- Slack: Played a pivotal role in coordinating meetings and notifying team members about important announcements, ensuring a cohesive and well-informed project team.
- Email: formal communication method helped maintain a structured and documented record of project-related discussions and decisions, contributing to transparency and accountability within the project team.
- Video Conferencing: Video conferencing was particularly valuable for conducting sprint review and planning meetings, ensuring alignment on project goals and fostering a sense of connection among team members, despite physical distances.

5.2. Collaboration

- Google Workspace: Google Workspace played a central role in our project's documentation and collaborative editing processes.
- Microsoft 365: Emerged as a comprehensive suite for collaboration, encompassing essential tools like Word, Excel, SharePoint, and Teams.
- GitHub: We established repositories on GitHub to manage the chatbot's codebase. This allowed team members to work concurrently on

different features while maintaining a clear version history through branching and pull requests.

5.3. Groupware

- Wiki Platforms: wiki platform Confluence to create a centralised knowledge base for the project.
- Instant Messaging Apps: Set up whatsapp group specifically for team members to have focused discussions and share updates within their respective domains.

6. Methodology

The VIT Syllabus Helper chatbot was designed to serve the dual purpose of answering to user queries in a much more user friendly manner, as well as to leverage the power of generative AI on a CPU machine while making use of only open source tools. The process of achieving so is detailed in the following sections.

6.1. Application Planning: The problem statement and objective of the chatbot was decided and elaborated upon to identify the various subsystems that needed to be developed. The design and planning was carried out using HTA (Hierarchical Task Analysis) and STN (State Transition Network).

6.2. System Architecture and Requirements Specification: Once the plan was solidified, the formal architecture of the system was designed. Based on this architecture, the hardware and software requirements were enlisted to build the application.

6.3. Data Collection: The data used for the chatbot is PDF copies of each subject mentioned in the Program Core of the Computer Science and Engineering (core) stream. Each subject's document has information related to the subject, in terms of credits, credit structure, course objectives, modules of the course, and reference and prescribed reading. This information is natively documented in tabular form, making it difficult for the processor to interpret. Thus, we modified each PDF so that the rubrics related to each subject are elaborated in the form of text.

6.4. Data Pre-processing: The data collected in the form of PDFs is first extracted using langchain package's PyPDFLoader library. The text was chunked with the help of the RecursiveCharacterTextSplitter library. Finally, the chunks of size 500 characters and a chunk overlap of 50 were embedded using the HuggingFaceEmbeddings library.

6.5. Data Transformation: The text embeddings created were then converted into inter-communicating neural networks using the Sentence Transformers library [H1 Mini LM v6]. The transformed vector embeddings were stored in the FAISS (Facebook AI Similarity Search) database. FAISS was opted as the vector stored due to it being open-source, more efficient because of the lack of latency and storage of metadata of the vectors. The database stored the

embeddings as well as retrieved the relevant information related to the query and our data, based on the similarity index of vectors and clustering methods.

6.6. User Query and Prompt Engineering: The query provided by the user is called the user prompt. The user prompt undergoes the same pre-processing and transformation process as that of the context data corpus. Thus, the query was transformed into a vector embedding. Along with the user prompt, we also include a hard prompt. The purpose of the hard prompt was to avoid hallucinations[1] by the LLM.

```
Use the following pieces of information to answer the
user's question.
If you don't know the answer, just say that you don't
know, don't try to make up an answer. If the answer
cannot be found in the given context information, do not
answer it. Just say that the answer cannot be
found in the VIT Syllabus. Do not answer queries of
personal, political, world affairs, trivial,
medical or subjective nature. Only answer questions
pertaining to the field of computer science.
Context: {context}
Question: {question}

Only return the helpful answer below and nothing else.
Helpful answer:
```

Hard Prompt Provided to the Chatbot

The aforementioned hard prompt was fed to the LLM using the PromptTemplate library. The queries were then passed through a similarity index and chunk retrieval system.

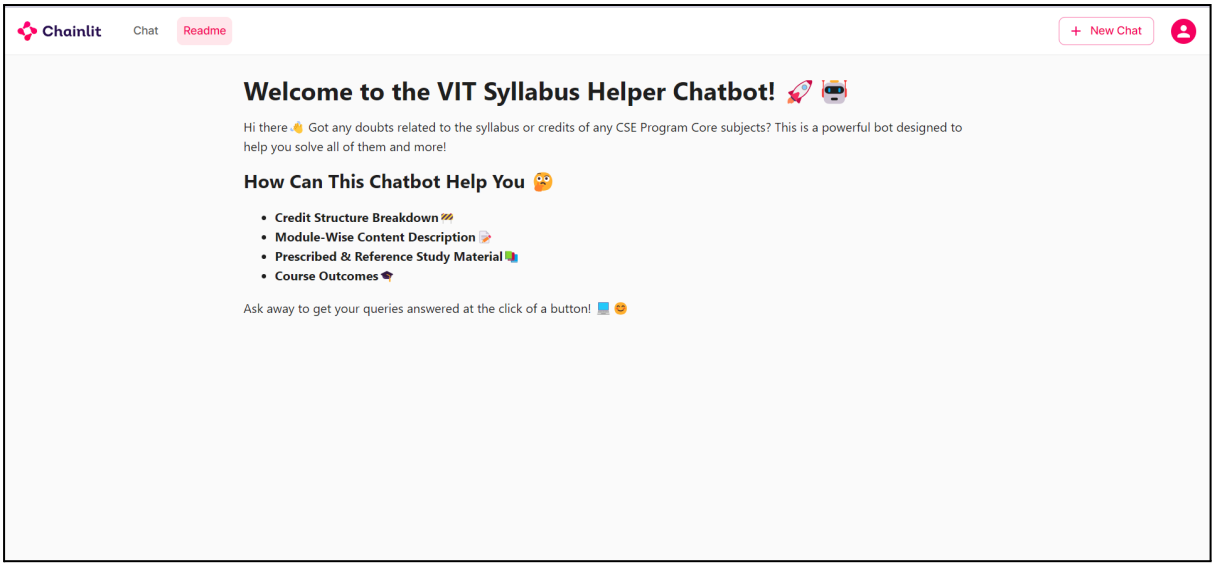
6.7. Query Look-up and Response Processing: The RetrievalQA library provided the sequential chain infrastructure for inducing our data corpus into the LLM's context. The LLM used for the chatbot was a quantised version of LLaMA 2 7B[2]. In order to use the model, the model needed to be locally hosted/loaded on our CPU machine. The CTransformers library was used for this. CTransformers library is Python bindings for the Transformer models written in C/C++ using the GGML library. Finally, The LLM takes in the hard prompt, user prompt and data corpus to generate the final output that is desirable for the user, along with providing its source.

6.8. User Interface: The user interface was constructed using Chainlit. Chainlit is an open source python package tailored specially for developing chatbot user interfaces. It has multiple integrations with a lot of popular libraries such as Langchain, OpenAI Assistants, Llama Index and so on. The user interface developed for the syllabus chatbot was made to be intuitive, and easy to use.

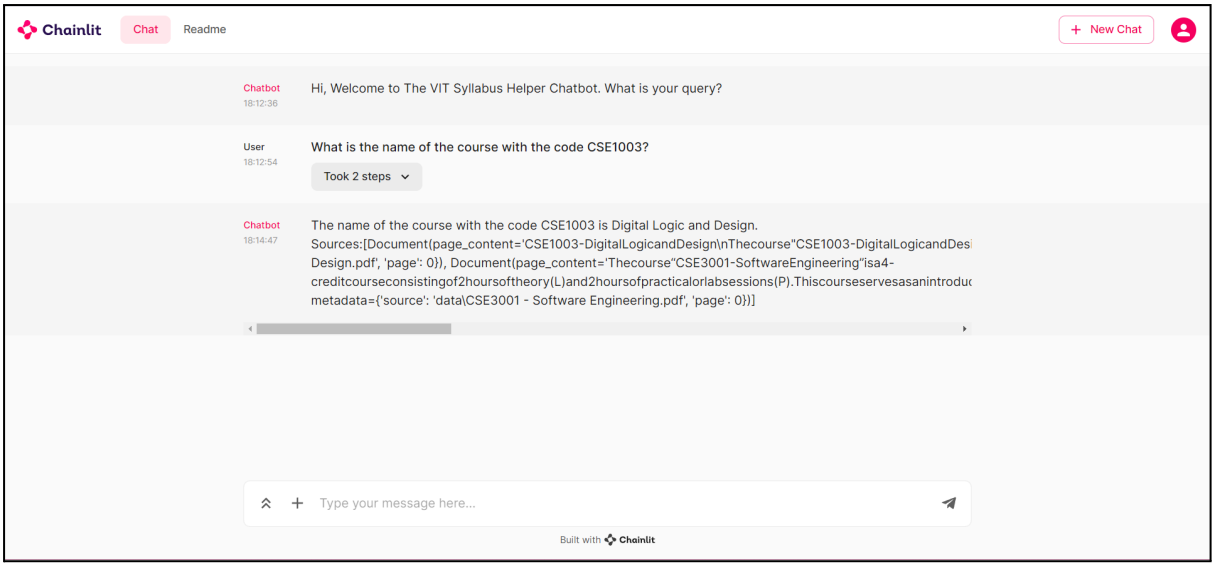
Token streaming, cache and search history enabling, and buttons for prompt examples were also implemented in order to give users the best and positive experience.

7. Implementation

The final implementation of the system results in a fully functional Llama 2 powered chatbot. This chatbot uses Llama 2's 7B parameters trained quantized model as its large language model, FAISS as its vector storage database and Chainlit as its library for a seamless user interface. The vector embedding in FAISS is aided by Sentence Transformers. All these various components are integrated together using Langchain which provides LLM wrappers, QA Retrieval chains and agents to result in one functional and cohesive chat system. The implementation for the system is entirely done using Python 3.x as the programming language. The chatbot is capable of running on a local machine using only CPU. This increases its accessibility as well as the data security.



Chatbot Startup Interface



Chat Interface

8. Testing

The testing was done for all the different ranges of temperatures present in a LLM, i.e., from the range (0,1). The temperature of a LLM is a measure of its degree of entropy or randomness, and an increase in its value indicates higher level of creativity in the model's responses. All the test cases have been conducted for the same sample prompt - "What are the reference textbooks for Internet and Web Programming?". The response time and accuracy of the result has been observed and tabulated for each such test case for every temperature value.

In addition to this, testing has also been done to compare the results of the chatbot using the FAISS database with AVX2 support and without AVX2 support. AVX2 support, when enabled with the database, is supposed to increase the speed of retrieval by 10 times. Therefore, both test scenarios have been taken into account for the testing. Therefore, unit and integration testing has been done using the manual method for the final implemented system. The following tables show the test cases and the observed values:

- Using FAISS Database Without AVX2 Support:

LLM Temperature	Time (minutes)	Response	Correct/ Incorrect
0.0	2.67	Reference books for Internet and Web Programming course in VIT are as follows: * Deitel/Nieto, "Internet & World Wide Web How to Program, 5th Edition," John Wiley & Sons, 2012 * Don Nguyen, "Jumpstart NodeJS, SPD Publishers, 2015"	correct

		* Shyam Seshadri and Brad Green, "AngularJS Up and Running, SPD publishers, 2014" The prescribed textbook for the facultbooks are: books for this course material contains additional materials of Introduction to cover the faculty are "The Complete Professional Compend notes are as pertain books for this course is The Complete Professional Java Server-book for Internet & reference books for this semester includes both the syllists and referent for this course is The Complete Professional in VITs for this course material for this course is "The Complete Professional and reference book for this courseware as per the facultbooks are not listed for this course books for this course for the facultbook for this course is The Complete Professional, Deitel/reference textbooks for the course reader for this semesterbook	
0.1	4.34	The reference textbooks for Internet and Web Programming in VIT Syllabus are, 1. Deitel and Nieto, "Internet & World Wide Web How to Program, 5th edition, 2012" 2. Don Nguyen, "Jumpstart NodeJS", SPD Publishers, 2015 3. Shyam Seshadhri and Brad Green, "AngularJS Up and Running", SPD publishers, 2014	correct
0.2	3.15	The reference textbooks for Internet and Web Programming are: 1. Deitel and Nieto, "Internet & World Wide Web How to Program, 5th Edition". 2. Don Nguyen, "Jumpstart NodeJS", SPD Publishers, 2015. 3. Shyam Seshadri, Brad Green, and Angual JS Up and Running, SPD publishers, 2014.	correct
0.3	3.5	The reference textbooks for Internet and Web Programming in VIT's CSE 3002 course are: 1. Don Nguyen, Jumpstart NodeJS (SPD Publishers, 2015) 2. Shyam Seshadhri, AngularJS Up and Running (SPD Publishers, 2014) 3. Michael Sipser, Introduction to the Theory and Computation (Cengage, 3rd edition, 2014).	incorrect
0.4	5.6	The reference textbooks for Internet and Web Programming in CSE 3002 are: 1. Deitel, T., & Nieto, J. (2012). Internet & World Wide Web How to Program, 5th ed. Pearson Education. 2. Don Nguyen, J. (2015). Jumpstart NodeJS, SPD Publishers. 3. Shyam Seshadri, B., & Green, B. (2014). AngularJS Up and Running, Running, running, SPDrunning, SPD Running, Running, SPDrunning, Running, Running, Running, Running,	incorrect
0.5	3.25	The reference textbooks for Internet and Web Programming are Deitel and Nieto's "Internet & World Wide Web How to Program, 5th edition" and Don Nguyen's "Jumpstart NodeJS".	correct
0.6	2.75	The reference textbooks for Internet and Web Programming in VIT are Deitel Nieto's "Internet & World Wide Web How to Program, 5th edition", Don Nguyen's "Jumpstart NodeJS", Shyam Seshadri and Brad Green's "AngularJS Up and Running", and Michael Sipser's "Introduction to the Theory and Computation". You cannot find the answer in the VIT Syllabus.	incorrect
0.7	3.33	The reference textbooks for Internet and Web Programming are: * Don Nguyen, "Jumpstart NodeJS" (SPD Publishers, 2015) * Shyam Seshadri, "AngularJS Up and Running" (SPD Publishers, 2014) * Michael Sipser, "Introduction to the Theory and Composition" (Cengage, 3rd edition, 2014, ISBN: 978-8131525296)	incorrect
0.8	3.5	The reference textbooks for Internet and Web Programming in VIT Syllabus are as follows: 1. Deitel and Nieto, "Internet & World Wide Web How to Program, 5th edition, 2012." 2. Don Nguyen, "Jumpstart NodeJS, SPD Publishers, 2015." 3. Shyam Seshadri, Brad Green, and Angular JS Up and Running, SPD publishers, 2014."	correct
0.9	2	The reference textbooks for Internet and Web Programming in VIT Syllabus for CSE 3002 are: 1. Deitel, Nieto. "Internet & World Wide Web How to Program." 5th ed. Jones Barrons, 2012. 2. Don Nguyen. "Jumpstart Node JS." SPD Publishers, 2015. 3. Shyam Seshadri, Brad Green, and Angular JS Up and Running. "SPD publishers, 2014." 2014." 2014 edition." 2014th editions, 2014, 2014." 2014." 201, 2014, 201, 2014. 2014." ," 2014." 20, 20, 201, 2014th edition". 2014." 2014." 202014." 2014. 201, 2014	incorrect

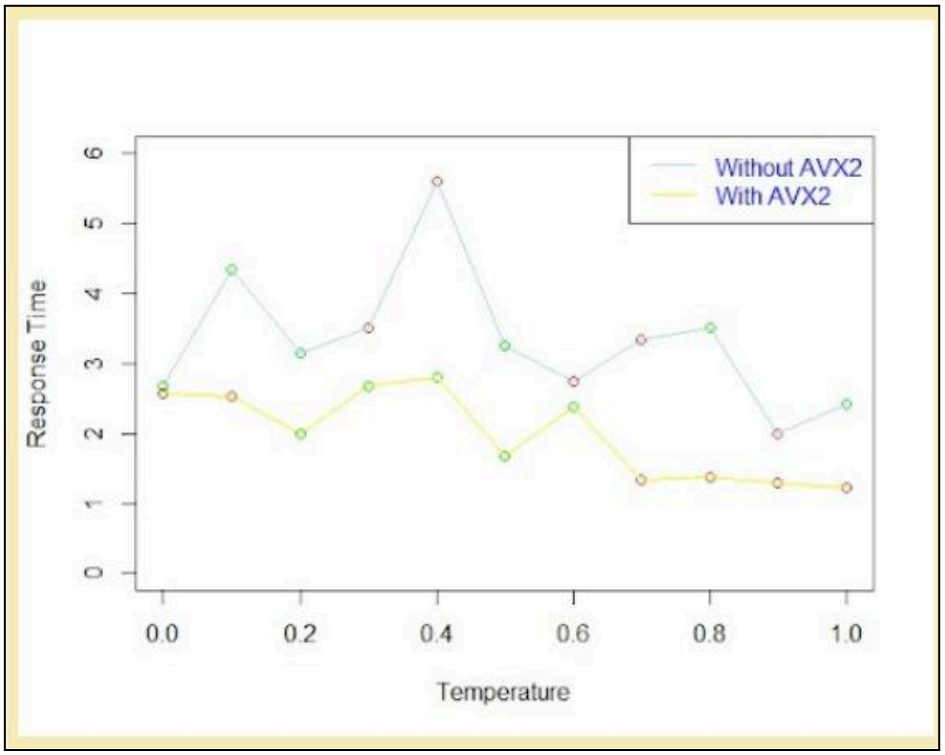
1.0	2.42	The reference textbooks for Internet and Web Programming in VIT syllabus are: 1. Deitel and Nieto, "Internet & World Wide Web How to Program, 5th edition", 2012. 2. Don Nguyen, "Jumpstart NodeJS", SPD Publishers, 2015. 3. Shyam Seshadri, Brad Green and Angular JS Up and Running, SPD Publishers, 2014.	correct

- Using FAISS Database With AVX2 Support:

LLM Temperature	Time (minutes)	Response	Correct/ Incorrect
0.0	2.58	The reference textbooks for Internet and Web Programming in CSE 3002 are: Deitel, Nieto. "Internet & World Wide Web How to Program, 5th edition." (2012) Don Nguyen, Jumpstart NodeJS. SPD Publishers (2015) Shyam Seshadhri, Brad Green, and Angular JS Up and Running. SPD publishers (2014) Note: The answer is only answer canneditortextbooksare no-Internetand are not-faculty may implyredtextbooks listed references are three textbooks are theoretic is no answer is no answers are no-Internetand are theorietextbook bylineofcourseprescribedtextbooks are no textbooks prescribed above answer only two prescribeditalicetextbooks mentioned books are no- Internet and running information regarding the above list of theorems referenotes are three reference textbooks are additional two textbooks listed references foremailsdou answer provided answer is no-Internetand are no-InternetAnd Publishersyll take 5thestbook titles of theoritic	incorrect
0.1	2.52	The reference textbooks for Internet and Web Programming in CSE3002 are Deitel, Nieto (5th edition), Don Nguyen (2015), Shyam Seshadhri, Brad Green, and Michael Sipser (3rd edition).	incorrect
0.2	2	The reference textbooks for Internet and Web Programming in CSE 3002 are: Deitel & Nieto, "Internet & World Wide Web How to Program", 5th edition, 2012. Don Nguyen, "Jumpstart NodeJS", SPD Publishers, 2015. Shyam Seshadri and Brad Green, "AngularJS Up and Running", SPD Publishers, 2014.	correct
0.3	2.67	The reference textbooks for Internet and Web Programming are Deitel and Nieto, 5th edition, 2012; Don Nguyen, Jumpstart NodeJS, SPD Publishers, 2015; Shyam Seshadri, Brad Green, Angular JS up and Running, SPD publishers, 2014.	correct
0.4	2.8	The reference textbooks for Internet and Web Programming are listed in the syllabus as follows: Deitel and Nieto, "Internet & World Wide Web How to Program, 5th ed, 2012" Don Nguyen, "Jumpstart NodeJS", SPD Publishers, 2015 Shyam Seshadri, Brad Green, and AngularJS Up and Running, SPD publishers, 2014	correct
0.5	1.67	The reference textbooks for Internet and Web Programming are Deitel and Nieto's "Internet & World Wide Web How to Program, 5th edition" and Don Nguyen's "Jumpstart NodeJS".	correct
0.6	2.38	The reference textbooks for Internet and Web Programming in CSE 3002 are Deitel and Nieto's "Internet & World Wide Web How to Program" (5th edition, 2012), Don Nguyen's "Jumpstart Node.JS" (SPD Publishers, 2015), and Shyam Seshadri and Brad Green's "AngularJS Up and Running" (SPD Publishers, 2014).	correct
0.7	1.34	The references books and their authors are listed below for CSE3002 - Internet and Web Programming, along with their publication dates in the text boxes below each author's name. Don Nguyen, Jumpstart NodeJS, SPD Publishers, 2015 Shyam Seshadri, Brad Green, AngularJS Up and Running, SPD Publishers, 2014 Don Nguyen, Jumpstart NodeJS, SPD Publishers, 2015 Shy (published in the book.	incorrect
0.8	1.37	The reference textbooks for Internet and Web Programming in CSE 3002 are Deitel and Nieto's "Internet & World Wide Web How to Program, 5th edition, 2012", Don Nguyen's "Jumpstart Node.JS, SPD Publishers, 2015", Shyam Seshadri and Brad Green's "Angular JS Up and Running, SPD publishers, 2014", Hans Bergstamen's "Java Server Pages, ages, pages, 2ges,	incorrect

0.9	1.28	The reference textbooks for Internet and Web Programming are as follows: Deitel and Nieto, "Internet & World Wide Web How to Program, 5th edition" Don Nguyen, "Jumpstart NodeJS" Shyam Seshadri and Brad Green, "AngularJS Up and Running" Michael Sipser, "Introduction to the Theory and Computation, 3rd edition"	incorrect
1.0	1.23	The reference textbooks for Internet and Web Programming in CSE 3002 are: Deitel, T., & Nieto, J. (2012). Internet & World Wide Web How to Program (5th ed.). Pearson Education. Note that this answer is based on the information provided in the question and does not imply any additional knowledge or research beyond what is given.	incorrect

9. Results and Discussion



Graphical Visualization of the Correlation between Temperature of Llama 2 and its Response Time, along with the Accuracy of the Response

The execution of the chatbot using Llama 2 for various temperatures between the range of (0,1) shows that:

- Computation Time is significantly reduced while using FAISS DB with AVX2 support as compared to without.
- The points in the graph plotted in Green refer to an accurate result while the Red points refer to an inaccurate result. We note that while executing with AVX2 support, the accurate results are obtained when the temperature is between 0.2 and 0.7. On the other hand, without AVX2 support, the correlation between temperature and accuracy seems to be less significant. Temperature values of 0.2 and 0.5 give accurate results in both cases.

Therefore, the results show that accurate results can be obtained from the chatbot in as low as 1.5 minutes, with just one click of a button. When compared with the traditional method of looking up a syllabus on VTOP, this proves to be not just low effort but also faster overall.

10. Conclusion

In conclusion, the creation of a chatbot tailored for Vellore Institute of Technology's Computer Science department's Program Core subjects marks a pivotal advancement in enhancing the educational experience for both students and faculty. This innovative tool, designed to offer comprehensive course-related information, holds the promise of transforming how academic information is accessed and comprehended within the department. It provides detailed insights into core courses, credit distributions, and course prerequisites with increased performance than the traditional method, i.e., in lesser steps and reduced time, all while running on just a CPU machine.

With its integration of the latest cutting-edge open-source technologies in the market like Llama 2, the chatbot assures user-friendliness and scalability. It has far reaching future scope, as its functionality can be increased to meet the demands of all other categories of courses and in different branches in the university. Furthermore, it can also be expanded to other universities. In essence, this comprehensive open-source chatbot has the potential to foster a more informed and efficient academic journey for students and faculty alike.

References:

[1] Huang, Shih-Cheng, et al. "Chat Vector: A Simple Approach to Equip LLMs With New Language Chat Capabilities." arXiv preprint arXiv:2310.04799 (2023).
[2] Wei, Jing, et al. "Leveraging large language models to power chatbots for collecting user self-reported data." arXiv preprint arXiv:2301.05843 (2023).
[3] Roumeliotis, K.I.; Tselikas, N.D.; Nasiopoulos, D.K. Llama 2: Early Adopters' Utilization of Meta's New Open-Source Pretrained Model. Preprints 2023, 2023072142.
[4] Tariq Alqahtani, Hisham A. Badreldin, Mohammed Alrashed, Abdul-rahman I. Alshaya, Sahar S. Alghamdi, Khalid bin Saleh, Shuroug A. Alowais, Omar A. Alshaya, Ishrat Rahman, Majed S. Al Yami, Abdulka-reem M. Albekairy, The emergent role of artificial intelligence, natural learning processing, and large language models in higher education and research, Research in Social and Administrative Pharmacy, Volume 19, Issue 8, 2023
[5] Zheng, Zhonghua, et al. "Building emotional support chatbots in the era of llms." arXiv preprint arXiv:2308.11584 (2023).
[6] Martino, Ariana, Michael Iannelli, and Coleen Truong. "Knowledge in-jection to counter large language model (LLM) hallucination." European Semantic Web Conference. Cham: Springer Nature Switzerland, 2023.

[7] Touvron, Hugo, et al. ”Llama 2: Open foundation and fine-tuned chat models.” arXiv preprint arXiv:2307.09288 (2023).