

**Course Title: Spring 2024 Course Practicum (CSP-572)**

**Report On:**

Project Proposal

**Submitted By:**

Albert Mandizha A20493341

Anette Volkova A20481653

Avinash Sankar A20525471

Chandana Poloju A20527866

Priyadarshini Rajendran A20476470

Ranjan Mishra A20521033

Vivian Jacob Varghese A20502106

Sahil Thorat A20543914

**Submitted To: Submitted Date:**

Zheng, Yong 01-24-2024

Computer Science Department

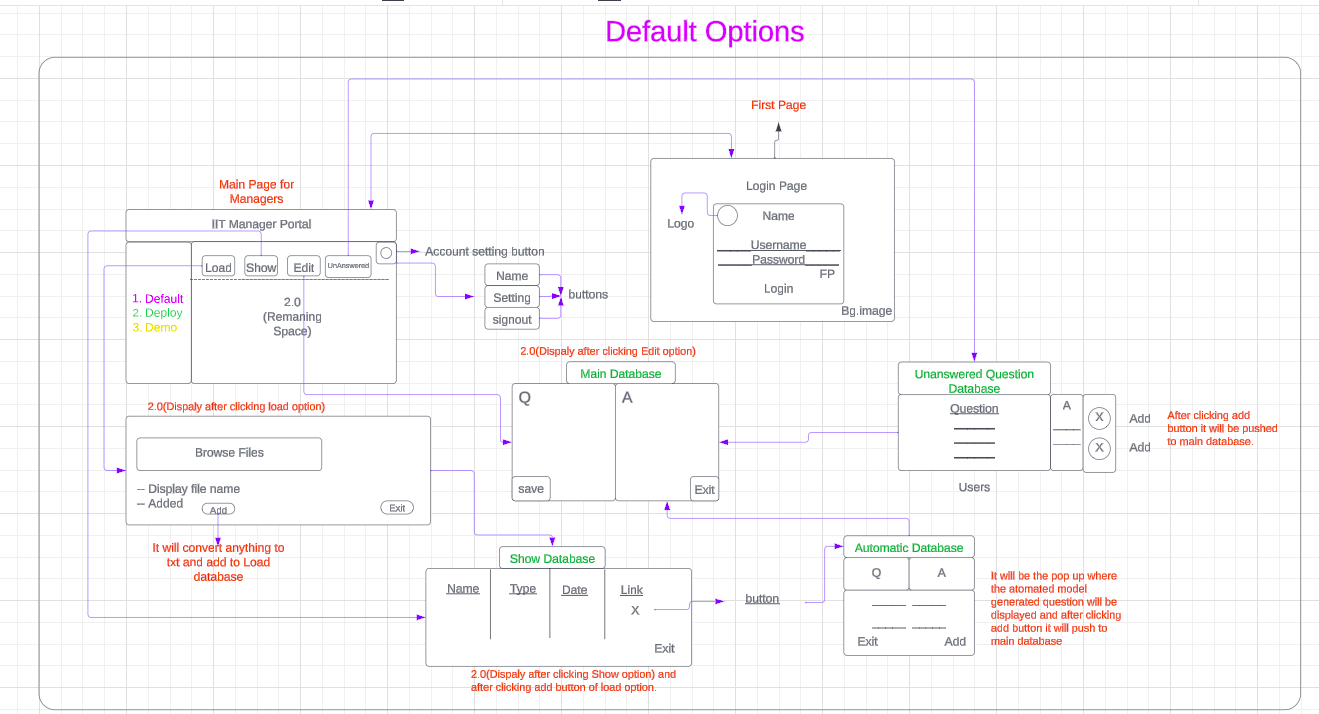
Illinois Institute of Technology

# Introduction

The goal of this project is to build a chatbot for the IIT Office of Research and Sponsored Programs (ORSP) which has the capabilities of answering questions for faculties. The ORSP office has a need to get quick responses about: grant applications; procedures pre, during, and after grant approvals; and other information regarding the office. The chatbot will ideally streamline this process. It will have the capability to utilize Google API for IIT credentials, allow for file uploads, allow for managers to make edits, storing multi-turn QA, and employ embedded systems.

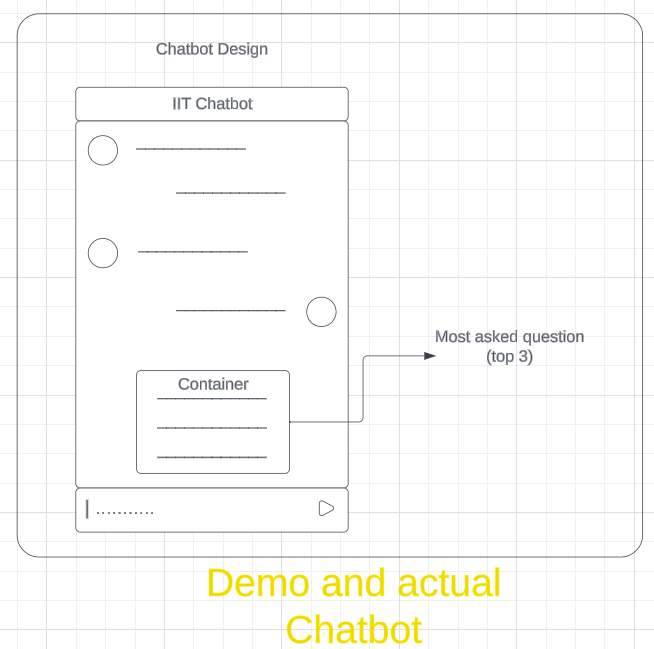
# System Architecture

For Manager:



1. Demo:

Chatbot interface/Also for users

Database:

1. Login credentials
2. Show Database
3. Edit main
4. Edit user
5. Link main

2. Deploy

# 

Programming: (Main)

1. Browse/Add files pdf, docx, url and convert them to text and add to the show database.
2. Link button (show database) activate. Automatically  
   Q+A are generated and stored in link main databases

^ Use of LLM

1. In link database manager can edit question and answer them accordingly and add them to exit main database.
2. Now for chatbot working when user will post question it will look for similar question from the edit main\* database and respond back the following answer. If not it will give default it answer add the question will be added to edit user database.

* Here we need to use again IR for converting questions + answers in vectors.

# Steps for Project

1. State the Objectives

* Have a system for managers where they can login/logout, change passwords, etc using Google API.
* The capability to upload a document to add QA entries into a database.
* Add, edit, change, and delete QA entries one by one in the database if desired.
* To store multi-turn QA and allow for managers to input them/design them.
* Save questions which were not answered by the chatbot.
* To be able to embed the chatbot on an IIT website or facebook
* To get clear answers to questions asked by users.
* To get the most accurate answers possible to questions asked by users.
* The responses to get answered in an acceptable amount of time.

1. Project Owner Requests/Expectations

* Create a chatbot to assist faculty with learning details about grant applications, grant documents, procedures pre, during and after grant approval using no GPU computing resources.

1. Client/Server(CS) System Development

* Utilizing python we hope to create the backend components of the client server system. We desire a clear, simple system which allows for editing of questions and answers to the database linked, as well as another to maintain unanswered questions which could be added later.

1. Chatbot Design

* A chatbot which has a clear path of question and answer. We hope to make it similar to the chatbots for the offices where Talon is responding to put a touch of school spirit on it.

1. Information Retrieval for QA Integration

* We hope to get the best accuracy possible with questions answered. We will utilize premade LLM models which match questions and answers using vector similarity.

Tasks split into sprints: (16 week period total)

Sprint 1: (week 1)

* Meet with the professor about desired outcomes as well as read thoroughly through project description.
* Break the project into steps.

Sprint 2: (week 2-3)

* Write the proposal and get the group on the same page.
* Set up the first database:
  + Develop basic authentication for managers in the CS system.
  + Develop the ability to add, edit, and delete QA entries in the database.
* Figure out how to store the multi-turn QA entries.

Sprint 3: (week 4-5)

* Create a second database for unanswered questions from the chat bot.
  + Create a way to add the questions and answers from the unanswered database to the main one.
* Connect the main database to the cs system.
* Create the beginning of the user interface.

Sprint 4: (week 6-7)

* Research chatbot tools with SQL integration capabilities.
* Build the chatbat
  + Test the connectivity to facebook and a website
* Connect second database to the main and CS system

Sprint 5: (week 8-10)

* Figure out how to convert QA entries into vectors.
* Use a pre-trained LLM for vector conversion.
* Theorize tests or certain trials that each individual component needs to pass and that the whole system needs to pass to ensure everything is working properly.
  + Run tests on each individual components.
  + Run tests on the whole system.

Sprint 6: (week 16)

* Monitor and improve if time permits.
* Finish cleaning and writing out work done on the whole system.

# Tools and Resources

Python

SQL

Flask

Open AI

Streamlit

MERN Stack

NLP Algorithms

# Challenges

1. Some initial challenges we ran into was figuring out the setup of this project, it was daunting trying to figure out all the pieces at once so we split the pieces up and worried about the connectivity of it all after we knew the functionality of each piece.
2. Sprint 2: How to connect the google API.
3. Sprint 2: How to store the multi-turn QA entries.
4. Sprint 5: Find a LLM which allows for good results in vector similarity results.

# 