Theme 2: Intelligent Learning Platform

Topic: Technical IT Learning Assistant

1. **What are the names and NetIDs of all your team members? Who is the captain? The captain will have more administrative duties than team members.**

   Single Team Member / Captain: Haitham Shahin – hshahin2@illinois.edu

2. **What topic have you chosen? Why is it a problem? How does it relate to the theme and to the class?**

   The topic for the project is to try to enhance learning for end-users by developing an MVP system that can answer queries from users based on technical documentation. This may include code snippets, documentation from GitHub readme's, or come from product documentation online.

   As a current cloud architect, much of my role when working with customers is educating them on how to use Azure cloud services. Specifically for larger enterprises however, this is not as straightforward as simply providing the online documentation from Azure. Companies often have their own internal procedures and technical documentation that dictate how one should go about accomplishing a task in IT. For example, companies often apply strict security controls to their Cloud deployments, so they capture these requirements in internal documents like GitHub Readme files or Internal Wiki pages. Thus, my vision is to build an intelligent app that can use these sources of information together (both external and internal documentation) to provide users with accurate instructions for how to accomplish an IT task that does account for company-specific details.

   This topic is very related to the course content. It will require the use of the search engine concepts we've learned to accurately select relevant documentation associated with the user's question. Additionally, language understanding and semantic understanding will be necessary to properly return steps for a task based on the user's question. The topic will incorporate generate models like GPT to further enhance the response.

3. **Briefly describe any datasets, algorithms or techniques you plan to use**

   Data will be sampled from the Azure documentation as example documents crawled and stored locally in the index. Additionally, sample GitHub Readme files will be generated and stored alongside the Azure documentation to create example documents that reflect company-specific technical documentation.

   The MVP solution will include the use of Azure Cognitive Search to form an index and provide relevance search queries and Azure OpenAI for a GPT language model to return a response back to the end-user in the form of instructions.

4. **How will you demonstrate that your approach will work as expected? Which programming language do you plan to use?**

   The solution will be demonstrated by using the sample documents to answer end-user questions provided from the application's front-end. A user will ask a question such as "How do I provision a

VM in a VNET" and the intelligent app will return the steps that take into account not just the online documentation, but also the company's internal documentation that may have specific steps as part of the procedure.

The solution will use Python as the backend and may also include a JavaScript front-end. The goal will be to deploy the app as a container, and in addition to the code there will be the use of Azure Cloud Services such as Azure Cognitive Search, Azure OpenAI, and Azure Storage.

5. **Please justify that the workload of your topic is at least 20*N hours, N being the total number of students in your team. You may list the main tasks to be completed, and the estimated time cost for each task.**

   The main tasks for this project include the following with an estimated number of hours for each task. In total, building out the MVP is expected to take around 80 hours at least with the following high level tasks:

   1. [2 Hours] Storing a sample set of Azure documents and GitHub Readme documents in Azure Storage
   2. [10 Hours] Creating an index on the documents using Azure Cognitive Search and testing to confirm that the index functions as expected
   3. [10 Hours] Creating an Azure OpenAI service that integrates with Cognitive Search
      1. This will likely require prompt engineering and a lot of testing to get this functioning with a higher degree of accuracy
   4. [20 Hours] Building a backend that calls Azure OpenAI with the end users query and returns a response
   5. [20 Hours] Building a frontend that enables end-users to ask their questions and integrates with the developed backend to return responses.
   6. [10 Hours] Containerization of frontend and backend
   7. [10 Hours] Documentation to deploy solution
   8. [20 Hours - Bonus] Infrastructure as Code to redeploy the solution in Azure with automation