DEVELOPMENT PROJECT

Project 1 – Restaurant Finding

Project overview

We will create an app (web app or mobile app) that use a Map API (can be OpenStreetMap API, Google Maps API, MapBox API) to provide users with information about restaurants near by.

Features:

- **User authentication:** Our app supports OAuth2 API so users can login using their Google accounts. The users have to login to use other app’s features.
- **Maps and markers:** Users can view a map at the current location or a map of one area (e.g. a city or a province) and many markers on the map that indicate the location of restaurants inside this area.
- **Search and filter:** Users can search for restaurants by their name, or their locations (e.g. within a distance from the current coordinates). The search result can be filtered by food types (e.g., noodle soup, sticky rice, spring rolls etc).
- **Place information:** When a user clicks on a marker, the following information will show up:
  - *Name of the place*
  - *Introduction to the place*
  - *List of available dishes and prices*
  - *Star rating* - show how many users have been posted for each star rating for the restaurant (eg., 4% for the 2 star ratings, 17% for 4 star rating etc).
- **Create a marker:** Users can click on the map to add a new restaurant into the database.
- **Update information:** When a user clicks on a marker, he can update the information about the place or delete the place from the database.

Requirements:

- **Map API and API key:** You can use OpenStreetMap API or Google Maps API or MapBox API. You will need to obtain an API key from your chosen Map API provider to use their services in your application.
- **Front-end:** HTML, CSS, and JavaScript, JSON, …
- **Database:** You can use a relational database engine like SQLite, MySQL or Postgresql.
- **Places data**: You have to collect data about many restaurants (it would be best if the places are inside Hanoi or Vietnam) and store the data in your database. You can use public data sources, such as OpenStreetMap, Google Maps, or collect your own data.
- **Hosting (for web ap):** There are several options available, such as Replit or Heroku.
- **Project's management:** During the development, tools such as Scrum, GitHub, Jira, and Trello are suggested to record members' efforts.

Project 2 - Writing Assistant

Project overview

We will create an app (web app or mobile app) that work as a writing assistant tool with features like grammar check, plagiarism check, text completion, and paraphrasing. The app will use API provied by OpenAI (https://platform.openai.com/docs/introduction) to perform the assigned tasks.

Features

- **User authentication:** Our app supports OAuth2 API so users can login using their Google accounts. The users have to login to use other app's features.
- **Input text:** An user can input some text (by typing or copy/paste) for further processing. With the input text, the user can continue with the following tasks:
- **Grammar check:** Check the input text for grammar and spelling errors. The app will highlight the errors and suggest some corrections.
- **Plagiarism checker:** Check the input text for plagiarism. The app will analyze the input text and compare it with other documents on the internet. The app will then provide a percentage of how much of the text matches other documents, as well as links to those documents.
- **Text completion:** Pick up where the user left off and complete the input text with generated sentences.
- **Paraphrasing:** Rephrase the input text. The app will provide several paraphrased versions of the text, allowing the user to choose the one that best fits their needs.
- **Dashboard:** Shows the activity history of the user by listing all input text that the user entered and the corresponding output.

Requirements:

- **OpenAI API key:** You will need to obtain an API Key from OpenAI. You can get one by signing up for an account.
- **Front-end:** HTML, CSS, and JavaScript, JSON, …

- **Database:** You can use a relational database engine like SQLite, MySQL or Postgresql.
- **Hosting (for web ap):** There are several options available, such as Replit or Heroku.
- **Project's management:** During the development, tools such as Scrum, GitHub, Jira, and Trello are suggested to record members' efforts.

Project 3 – Online practical system for HANU Test

Project overview

We will create an app (web app or mobile app) that work as an examination preparation system for the output foreign language exam - HANU Test. Students can use this system to practice for a Vietnamese language exam with a sample test from:

https://tec.hanu.vn/c/12602/De-thi-minh-hoa-nang-luc-Tieng-Viet-dung-cho-nguoi-nuoc-ngoai

Features

- **User authentication:** Our app supports OAuth2 API so users can login using their Google accounts. The users have to login to use other app's features.
- **Test for reading skill:** Sample questions and test pages for reading skill
- **Test for listenting skill:** Sample questions and test pages for listening skill.
- **Test for writing skill:** Sample questions and test pages for writing skill.
- **Test for speaking skill:** Sample questions and test pages for speaking skill.
- **Dashboard:** Shows the activity history and testing result of the user.

Requirements:

- **Front-end:** HTML, CSS, and JavaScript, JSON, …
- **Database:** You can use a relational database engine like SQLite, MySQL or Postgresql.
- **Hosting (for web ap):** There are several options available, such as Replit or Heroku.
- **Project's management:** During the development, tools such as Scrum, GitHub, Jira, and Trello are suggested to record members' efforts.

RESEARCH PROJECT

Project 4 – Subject Tutor Chatbot

Project overview

The Subject Tutor Chatbot is a ChatGPT-based chatbot that provides guidance and support to students who are struggling with their studies in a particular subject. The chatbot is designed to help students get quick answers to their questions related to their studies in the subject. The chatbot is a kind of personalized ChatGPT chatbot trained on the subject's dataset.

Features:

- **Natural Language Processing:** The chatbot uses NLP algorithms to understand natural language queries and provide relevant answers to students.
- **Study Guidance:** The chatbot provides guidance and support to students who are struggling with their studies.
- **Frequently Asked Questions (FAQs):** The chatbot can provide answers to frequently asked questions about academic programs, courses, and resources.
- **Study Resources:** The chatbot can provide students with study resources like books, articles, and videos related to their subjects.
- **Study Tips:** The chatbot can provide students with study tips to help them improve their grades.

Project Requirements:

- **Front-end:** HTML, CSS, and JavaScript, JSON, …
- **Chatbot Framework and models:** You should choose a chatbot framework like:
  - *Vietnamese:* Underthesea (https://github.com/undertheseanlp/underthesea), PhoBert (https://github.com/VinAIResearch/PhoBERT) + Rasa (https://rasa.com/).

  - *English*: ChatterBot(https://chatterbot.readthedocs.io/en/stable/) or NLTK (https://www.nltk.org/).

- **Database:** You have determined the data you need to store (information about the students, their academic performance, their study preferences, and the resources they need). You can research existing databases that contain similar data or you can build your own database. To populate your database, you can collect data from students through surveys, questionnaires, or interviews. You can ask them about their academic performance, study habits, and preferences for study resources.
- **Hosting:** There are several options available, such as Replit, Pythonanywhere or Heroku.
- **Reference work:** http://lib.uet.vnu.edu.vn/bitstream/123456789/1199/1/Lu%E1%BA%ADn%20V%C4%83n%20-%20B%C3%B9i%20Ng%E1%BB%8Dc%20Quang%2013.06.pdf