user_cli.py

user_cli.py

This file is an older implementation of auto anki via Terminal

```
import shutil
import sys
import concurrent.futures
import pyfiglet
from extract_sizes import extract_words, text_to_groupings
import wordprocessing as wp
from google_search import get_people_also_ask_links
from anki import add_question, get_deck, get_model, add_package


def user_menu():
```

```
  Runner class. Prompts the user for input and returns a txt file of results
```

```
    format_welcome_message = pyfiglet.figlet_format("AUTO ANKI")
    size = shutil.get_terminal_size(fallback=(120, 50))
    valid_choices = ["1", "2", "Q", "q"]
    print(format_welcome_message.center(size.columns) + "\n")
    print("Welcome to Lecture Aid. Choose from the following options:\n")
    print("Option 1: Press 1 to enter the file location you "
          "would like Lecture Aid to help you find resources on.")
    print("Option 2: Press 2 ")
    print()
    print("Press Q to quit the program.")

    while True:
        choice = input("Please Enter your choice:")[0]
        if choice in valid_choices:
            break

        print("That choice is not available now. Please try again")
        continue

    if choice == valid_choices[0]:
        file_path = input("Please enter the path to the file: ")
        deck_name = input("Please enter the name of the lecture: ")
        return file_path, deck_name

    if choice == valid_choices[1]:
        input("")

    elif choice in [valid_choices[-1], valid_choices[-2]]:
        print("Thank you for using Auto Anki. Closing Program now.")
        sys.exit(0)


if __name__ == "__main__":
    file, lect_name = user_menu()
```

if file.endswith(“.pdf”): raw_data = extract_words(file) if file.endswith(“.docx”): raw_data = extract_words_word(file)

```
    raw_data = extract_words(file)
    raw_data = text_to_groupings(raw_data)
    keyword_data = wp.extract_noun_chunks(raw_data)
    keyword_data = wp.merge_slide_with_same_headers(keyword_data)

    keyword_data = wp.duplicate_word_removal(keyword_data)
    search_query = wp.construct_search_query(
        keyword_data)
    with concurrent.futures.ThreadPoolExecutor(max_workers=10) as executor:
```

when testing use searchquery[:10 or less]. Still working on better threading to get faster results

```
        results = executor.map(get_people_also_ask_links, search_query[:3])

    auto_anki_model = get_model()
    deck = get_deck(deck_name=lect_name)
```

```python
        for result in results:
            for qapair in result:
                question = qapair["Question"]
                answer = qapair["Answer"]
                qa = add_question(
                    question=f'{question}', answer=f'{answer}', curr_model=auto_anki_model)
                deck.add_note(qa)

        add_package(deck, lect_name)
```