

user_cli.py

MIT License

Copyright 2023 auto_anki

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE. user_cli.py

This file is an older implementation of auto anki via Terminal

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

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

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

```
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():

    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()

    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:

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

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

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

add_package(deck, lect_name)
```