



**MANIPAL INSTITUTE OF TECHNOLOGY**  
**BENGALURU**  
*(A constituent unit of MAHE, Manipal)*

## **WSMA MINI PROJECT REPORT**

On

### **GitHub Profile Analyzer**

**Submitted By:**

Varikuti Mahathi Reddy

235816142

**Submitted To:**

Ms. Charu Chauhan

Assistant Professor, School of Computer Engineering

**School of Computer Engineering**

Date of Submission: 20/10/2025

## **Abstract**

The Personal GitHub Profile Analyzer is an interactive web dashboard designed to provide a comprehensive, data-driven overview of any public GitHub user's profile. In the modern software development landscape, a GitHub profile serves as a dynamic resume. However, manually assessing a user's skills, primary interests, and activity level is often time-consuming and subjective. This project addresses that gap by creating a tool that automates the analysis process.

Using Python and the Streamlit framework, the application fetches data directly from the official GitHub API. It processes this data to generate key insights, including a user's most frequently used programming languages, their most popular repositories by star count, and general account statistics. A key feature is the application of Natural Language Processing (NLP) through the TF-IDF algorithm to analyze the textual content of repository README files, extracting the most significant project keywords.

The final output is a clean, interactive, and easily digestible dashboard that visualizes these metrics, offering a quick yet powerful summary of a developer's technical footprint. The project successfully demonstrates the integration of external APIs, data analysis, NLP techniques, and web-based visualization.

## **1. Introduction**

For software developers, recruiters, and technical managers, a GitHub profile has become a critical tool for showcasing technical abilities and project experience. It acts as a living portfolio, offering a transparent view into a developer's coding habits, interests, and contributions to the open-source community. While valuable, interpreting this information quickly and objectively presents a challenge. Assessing a profile often involves manually browsing through repositories, reading code, and trying to piece together a coherent picture of the user's expertise.

This project was initiated to streamline this process by creating an automated tool that generates a concise and insightful summary of any public GitHub profile. The "Personal GitHub Profile

Analyzer" is a web-based dashboard that transforms raw profile data into actionable insights, making it easier to understand a developer's technical strengths and areas of focus at a glance.

## 2. Problem Statement

The primary problem is the lack of a simple, accessible tool for the rapid, data-driven analysis of a GitHub user's public profile. Key stakeholders face the following challenges:

- **Developers:** Find it difficult to objectively assess the focus of their own profile or that of potential collaborators.
- **Recruiters:** Spend significant time manually reviewing profiles to gauge a candidate's primary skills and experience.
- **Managers:** Need a quick way to understand the technical expertise of team members or new hires.

Existing methods are manual, subjective, and do not efficiently aggregate key metrics like language proficiency, project popularity, or thematic focus. This project aims to solve this problem by providing an automated, visual, and quantitative summary.

## 3. Objectives

The primary objectives of this project are as follows:

- To develop a user-friendly, interactive web dashboard using the Streamlit framework.
- To ethically fetch public user and repository data from the official GitHub API.
- To analyze and aggregate the programming languages used across a user's repositories.
- To identify and rank a user's top repositories based on community engagement (star count).
- To apply Natural Language Processing (TF-IDF) to perform keyword extraction on repository README files, identifying a user's main project themes.
- To visualize all analyzed data in a clean, consolidated dashboard.
- To provide a feature for downloading a text-based summary of the generated report.

## 4. Scope

The scope of this project is defined by the following boundaries:

### In Scope:

- Analysis of public GitHub user profiles and their public repositories only.
- Data fetching is limited to the information provided by the public GitHub REST API.
- Text analysis is confined to the content of repository README files.
- Visualization includes top 5 programming languages and top 5 repositories.
- Keyword extraction is limited to the top 15 most relevant terms.

### Out of Scope:

- Analysis of private repositories or organizational data.
- Analysis of commit history, contribution frequency, or code complexity.
- User authentication (the application uses a developer's API token on the backend for rate-limiting purposes only).
- Sentiment analysis of issue comments or pull request discussions.

## 5. System Design / Methodology

The project follows a simple, modular architecture designed to separate data collection, analysis, and presentation.

### 5.1. Requirements

- **Functional:**
  - The system must provide an input field for a GitHub username.
  - It must display the user's avatar, name, and key statistics (follower count, etc.).
  - It must display a bar chart of the top programming languages.
  - It must list the top repositories by star count with links.
  - It must display a list of extracted project keywords.

- It must provide a button to download the report as a .txt file.
- **Non-Functional:**
  - The application must be responsive and provide feedback (e.g., a loading spinner) during data fetching.
  - The dashboard must be clean, readable, and present information clearly.
  - API calls should be efficient to minimize load times.

## 5.2. Design (Architecture)

The application follows a simple data flow:

1. **User Interface (Streamlit Frontend):** The user enters a GitHub username and clicks "Analyze".
2. **Backend Logic (Python/Streamlit):**
  - The request triggers the main analysis function.
  - **Data Collection:** Helper functions make sequential GET requests to the GitHub API to fetch user data, repository lists, and the raw text of README files.
  - **Data Processing & Analysis:** The fetched JSON data is parsed. The code aggregates language data, sorts repositories, and performs text cleaning and TF-IDF analysis on the README content.
  - **Visualization:** A Matplotlib chart is generated from the language data.
3. **User Interface (Streamlit Frontend):** The Streamlit app dynamically renders the processed data, tables, and the Matplotlib chart on the dashboard.

## 5.3. Tools & Technologies

- **Python:** The core programming language for the entire application.
- **Streamlit:** The web framework used to build the interactive dashboard with Python-only code.

- **Requests:** The library used for making HTTP requests to the GitHub API.
- **Pandas:** Used for organizing repository data into a structured DataFrame for easy display.
- **Scikit-learn:** Used for its TfidfVectorizer to perform keyword extraction.
- **Matplotlib:** The library used to generate the language distribution bar chart.
- **GitHub API:** The external REST API used as the primary data source.

## 6. Implementation

The project is implemented in a single Python script (app.py) using the Streamlit framework.

The code is organized into several key functional areas:

- **API Helper Functions (get\_user\_data, get\_repos, get\_readme\_content):** Each function is responsible for a specific API call, handling authentication headers and response parsing. This modularity makes the code easy to debug.
- **Analysis Functions (analyze\_languages, get\_top\_repos, extract\_keywords):** These functions take the raw data from the API helpers and perform specific analytical tasks. The extract\_keywords function includes a text cleaning pipeline using regular expressions to remove URLs, HTML tags, and other noise before analysis.
- **Visualization Function (create\_language\_chart):** This function takes the aggregated language data and uses Matplotlib to generate a styled bar chart, which is then passed to the Streamlit frontend.
- **Streamlit UI:** The final section of the code uses Streamlit's commands (st.title, st.text\_input, st.columns, st.pyplot, st.dataframe) to construct the entire user interface and orchestrate the calls to the analysis functions.

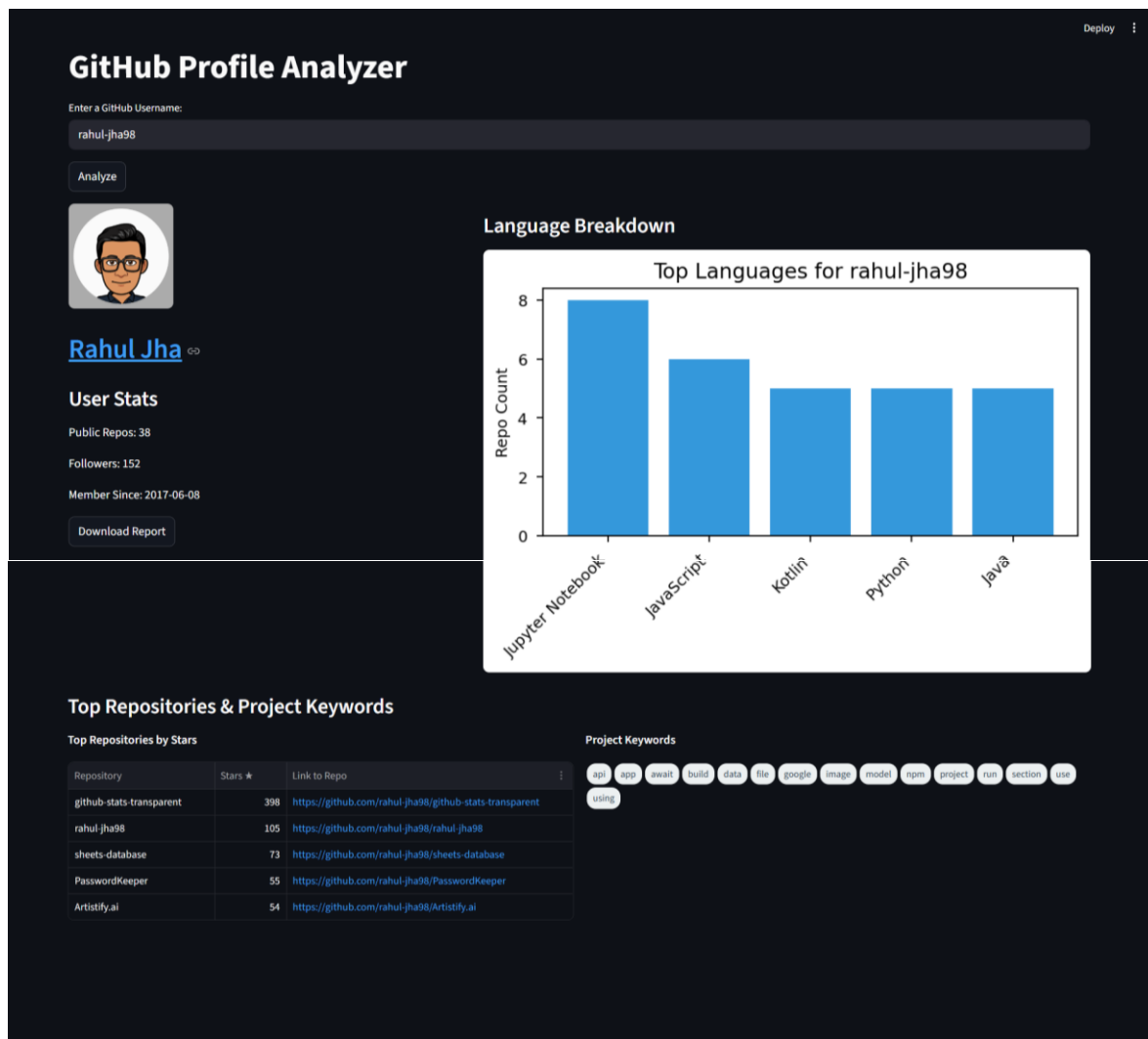
## 7. Testing and Results

The application was tested with various GitHub usernames to ensure robustness, including profiles with many repositories (e.g., google), profiles with few repositories, and profiles with non-English READMEs. The error handling for non-existent users was also verified.

## Results:

The final application successfully generates a dashboard as intended. For a sample user, the dashboard provides a clear overview:

- The **User Stats** section gives an immediate sense of the user's scale and tenure.
- The **Language Breakdown** chart quickly identifies the user's primary tech stack.
- The **Top Repositories** table highlights their most impactful work.
- The **Project Keywords** section provides a deeper, thematic insight into their specific interests and expertise (e.g., AI, web development, data science).



## 8. Discussion

The project successfully demonstrates that a great deal of insight can be derived from the GitHub API. The combination of quantitative data (like star counts and language usage) with qualitative data (keyword analysis from READMEs) provides a well-rounded view of a developer's profile.

One key finding during development was the necessity of a robust text-cleaning pipeline for the keyword extraction. Initial versions produced noisy keywords from URLs, image links, and code snippets. The implementation of a multi-step cleaning process using regular expressions significantly improved the quality and relevance of the extracted keywords.

A limitation of the project is its reliance on well-maintained README files for keyword analysis. Users with sparse or empty READMEs will yield less insightful keyword results.

## 9. Conclusion and Future Work

### Conclusion:

This project successfully achieved its objective of creating a functional, interactive dashboard for analyzing GitHub profiles. It serves as a valuable tool for developers, recruiters, and managers by automating the process of data collection and presenting insights in a clear, visual format. The use of Streamlit allowed for rapid development and iteration, resulting in a polished and professional-looking application.

### Future Work:

The project has several avenues for future enhancement:

- **Commit Activity Analysis:** Integrate analysis of commit history to visualize a user's contribution frequency and patterns over time.
- **Contribution Graph:** Recreate the iconic GitHub contribution graph with more detailed statistics.
- **User Comparison:** Add a feature to analyze two user profiles side-by-side.



- **Deployment:** Deploy the application to a cloud service like Streamlit Community Cloud to make it publicly accessible.

## 10. References

- **Streamlit (Framework):** Streamlit Inc. (2023). *Streamlit Documentation*. Retrieved October 20, 2025, from <https://docs.streamlit.io/>
- **GitHub API (Data Source):** GitHub, Inc. (2023). *GitHub REST API Documentation*. Retrieved October 20, 2025, from <https://docs.github.com/en/rest>
- **Scikit-learn (NLP Library):** Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830.
- **Matplotlib (Visualization Library):** Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90-95.
- **Pandas (Data Manipulation Library):** The pandas development team. (2023). *pandas-dev/pandas: Pandas*. Zenodo. <https://doi.org/10.5281/zenodo.8282384>

# 11. Appendices

## Appendix A: Source Code

```
1 import streamlit as st
2 import requests
3 from collections import Counter
4 import matplotlib.pyplot as plt
5 from sklearn.feature_extraction.text import TfidfVectorizer
6 import pandas as pd
7 import re  # Import the regular expressions library
8
9 # Import your token from config.py
10 from config import GITHUB_TOKEN
11
12 # ... Global API helper functions (changes from original) ...
13 def get_repo_details(repo):
14     """Fetch basic repo data"""
15     url = f'https://api.github.com/repos/{username}/{repo}'
16     headers = {'Authorization': f'token {GITHUB_TOKEN}'}
17     response = requests.get(url, headers=headers)
18     if response.status_code == 200:
19         return response.json()
20     return None
21
22 def get_repos(username):
23     """Getting all public repositories for a user"""
24     repos = []
25     page = 1
26     while True:
27         url = f'https://api.github.com/users/{username}/repos?page={page}'
28         headers = {'Authorization': f'token {GITHUB_TOKEN}'}
29         response = requests.get(url, headers=headers)
30         if response.status_code == 200:
31             data = response.json()
32             if not data:
33                 break
34             repos.extend(data)
35             page += 1
36         else:
37             return None
38     return repos
39
40 def get_repo_content(repo_full_name):
41     """Fetch the README content for a single repository"""
42     url = f'https://api.github.com/repos/{repo_full_name}/readme'
43     headers = {
44         'Authorization': f'token {GITHUB_TOKEN}',
45         'Accept': 'application/vnd.github.v3+json'
46     }
47     response = requests.get(url, headers=headers)
48     if response.status_code == 200:
49         return response.text
50     return None
51
52 # ... Analysis functions ...
53 def analyze_languages(repos):
54     """Counts the primary language for each repo"""
55     if not repos:
56         return None
57     lang_counter = Counter([repo['language'] for repo in repos if repo['language']])
58     return lang_counter.most_common()
59
60 def get_top_repos(repos):
61     """Returns the top 5 repos by star count"""
62     if not repos:
63         return []
64     sorted_repos = sorted(repos, key=lambda x: x['stargazers_count'], reverse=True)
65     return sorted_repos[:5]
66
67 def extract_keywords(repos):
68     """Clean README text and extract top keywords using TF-IDF"""
69     readme = [get_repo_content(repo['full_name']) for repo in repos if not repo['fork']]
70     if not readme:
71         return []
72
73     cleaned_readme = []
74     for text in readme:
75         # 1. Remove HTML
76         text = re.sub('<.*>', '', text, flags=re.MULTILINE)
77         # 2. Remove single tags for whitespace cleanup
78         text = re.sub('<\/?br>', ' ', text)
79         # 3. Remove HTML tags
80         text = re.sub('<.*>', '', text)
81         # 4. Remove non-alphanumeric characters (but keep spaces)
82         text = re.sub('[^a-zA-Z ]+', '', text)
83         # 5. Convert to lowercase
84         text = text.lower()
85     cleaned_readme.append(text)
86
87     if not any(cleaned_readme):
88         return []
89
90     vectorizer = TfidfVectorizer(stop_words='english', max_features=10)
91     try:
92         vectorizer.fit_transform(cleaned_readme)
93     except ValueError:
94         return vectorizer.get_feature_names_out().tolist()
95
96 # This can happen if all text is filtered out
97 return []
98
99 # ... Visualization function (modified for Streamlit) ...
100 def create_language_chart(lang_data, username):
101     """Creates a bar chart and returns the Matplotlib figure"""
102     if not lang_data:
103         return None
104
105     languages_counts = [x[0] for x in lang_data]
106
107     plt.style.use('seaborn-v0.8-paper')
108     fig, ax = plt.subplots(figsize=(8, 5))
109     ax.bar(languages_counts, color='lightcoral')
110     ax.set_title('Top Languages for {username}'.format(username=username), fontsize=12)
111     ax.set_xlabel('Language', fontsize=10)
112     plt.xticks(rotation=45, ha='right', fontsize=9)
113     plt.tight_layout()
114     plt.show()
115
116 # ... Report generation function ...
117 def generate_report_text(user_data, repos, lang_data, top_repos, keywords):
118     report = f'Github Repository Analysis Report for {user_data["login"]} user data logs [1/1]'
119     report += '\n\n'
120     report += f'Public Repos: {len(repos)}'
121     report += f'Followers: {user_data["followers"]}'
122     report += f'Public Issues: {len([issue for issue in user_data["issues"] if issue["public"]])}'
123     report += f'Top 5 Repositories by Stars: {top_repos}'
124     report += '\n\n'
125     report += f'Top 5 Programming Languages: {lang_data}'
126     report += '\n\n'
127     report += f'Top 5 Project Keywords: {keywords}'
128     return report
129
130 # ... Streamlit UI ...
131 st.set_page_config(layout='wide')
132 st.title('Personal Github Profile Analyzer')
133
134 username = st.text_input('Enter a Github username (e.g., "octocat")')
135
136 if st.button('Analyze'):
137     if not username:
138         st.warning('Please enter a username.')
139     else:
140         st.session_state['username'] = username
141         st.session_state['repos'] = get_repos(username)
142         st.session_state['lang_data'] = analyze_languages(repos)
143         st.session_state['top_repos'] = get_top_repos(repos)
144         st.session_state['keywords'] = extract_keywords(repos)
145
146         # ... Build the dashboard ...
147         col1, col2 = st.columns(2)
148
149         with col1:
150             st.subheader('User Profile')
151             st.json(user_data)
152             st.subheader('Public Repositories')
153             st.json([repo['full_name'] for repo in repos])
154
155         with col2:
156             st.subheader('Languages')
157             create_language_chart(st.session_state['lang_data'], username)
158
159         st.subheader('Top Repositories & Project Keywords')
160         st.json([repo['full_name'] for repo in top_repos])
161         st.json([keyword for keyword in keywords])
162
163         st.button('Generate Report')
164         report_text = generate_report_text(user_data, repos, lang_data, top_repos, keywords)
165
166         st.download_button(
167             label='Download Report',
168             data=report_text,
169             file_name=f'{username}_report.txt',
170             mime='text/plain'
171         )
172
173         # ... Streamlit UI ...
174         st.subheader('Project Keywords')
175         st.json(keywords)
176
177         # ... Streamlit UI ...
178         st.subheader('Public Repositories')
179         st.json([repo['full_name'] for repo in repos])
180
178 # ... Streamlit UI ...
179 st.subheader('Project Keywords')
180 st.json(keywords)
181
182 # ... Streamlit UI ...
183 st.subheader('Public Repositories')
184 st.json([repo['full_name'] for repo in repos])
185
186 # ... Streamlit UI ...
187 st.subheader('Project Keywords')
188 st.json(keywords)
189
190 # ... Streamlit UI ...
191 st.subheader('Public Repositories')
192 st.json([repo['full_name'] for repo in repos])
193
194 # ... Streamlit UI ...
195 st.subheader('Project Keywords')
196 st.json(keywords)
197
198 # ... Streamlit UI ...
199 st.subheader('Public Repositories')
200 st.json([repo['full_name'] for repo in repos])
201
202 # ... Streamlit UI ...
203 st.subheader('Project Keywords')
204 st.json(keywords)
205
206 # ... Streamlit UI ...
207 st.subheader('Public Repositories')
208 st.json([repo['full_name'] for repo in repos])
209
210 # ... Streamlit UI ...
211 st.subheader('Project Keywords')
212 st.json(keywords)
213
214 # ... Streamlit UI ...
215 st.subheader('Public Repositories')
216 st.json([repo['full_name'] for repo in repos])
217
218 # ... Streamlit UI ...
219 st.subheader('Project Keywords')
220 st.json(keywords)
221
222 # ... Streamlit UI ...
223 st.subheader('Public Repositories')
224 st.json([repo['full_name'] for repo in repos])
225
226 # ... Streamlit UI ...
227 st.subheader('Project Keywords')
228 st.json(keywords)
229
230 # ... Streamlit UI ...
231 st.subheader('Public Repositories')
232 st.json([repo['full_name'] for repo in repos])
233
234 # ... Streamlit UI ...
235 st.subheader('Project Keywords')
236 st.json(keywords)
237
238 # ... Streamlit UI ...
239 st.subheader('Public Repositories')
240 st.json([repo['full_name'] for repo in repos])
241
242 # ... Streamlit UI ...
243 st.subheader('Project Keywords')
244 st.json(keywords)
245
246 # ... Streamlit UI ...
247 st.subheader('Public Repositories')
248 st.json([repo['full_name'] for repo in repos])
249
250 # ... Streamlit UI ...
251 st.subheader('Project Keywords')
252 st.json(keywords)
253
254 # ... Streamlit UI ...
255 st.subheader('Public Repositories')
256 st.json([repo['full_name'] for repo in repos])
257
258 # ... Streamlit UI ...
259 st.subheader('Project Keywords')
260 st.json(keywords)
261
262 # ... Streamlit UI ...
263 st.subheader('Public Repositories')
264 st.json([repo['full_name'] for repo in repos])
265
266 # ... Streamlit UI ...
267 st.subheader('Project Keywords')
268 st.json(keywords)
269
270 # ... Streamlit UI ...
271 st.subheader('Public Repositories')
272 st.json([repo['full_name'] for repo in repos])
273
274 # ... Streamlit UI ...
275 st.subheader('Project Keywords')
276 st.json(keywords)
277
278 # ... Streamlit UI ...
279 st.subheader('Public Repositories')
280 st.json([repo['full_name'] for repo in repos])
281
282 # ... Streamlit UI ...
283 st.subheader('Project Keywords')
284 st.json(keywords)
285
286 # ... Streamlit UI ...
287 st.subheader('Public Repositories')
288 st.json([repo['full_name'] for repo in repos])
289
290 # ... Streamlit UI ...
291 st.subheader('Project Keywords')
292 st.json(keywords)
293
294 # ... Streamlit UI ...
295 st.subheader('Public Repositories')
296 st.json([repo['full_name'] for repo in repos])
297
298 # ... Streamlit UI ...
299 st.subheader('Project Keywords')
300 st.json(keywords)
301
302 # ... Streamlit UI ...
303 st.subheader('Public Repositories')
304 st.json([repo['full_name'] for repo in repos])
305
306 # ... Streamlit UI ...
307 st.subheader('Project Keywords')
308 st.json(keywords)
309
310 # ... Streamlit UI ...
311 st.subheader('Public Repositories')
312 st.json([repo['full_name'] for repo in repos])
313
314 # ... Streamlit UI ...
315 st.subheader('Project Keywords')
316 st.json(keywords)
317
318 # ... Streamlit UI ...
319 st.subheader('Public Repositories')
320 st.json([repo['full_name'] for repo in repos])
321
322 # ... Streamlit UI ...
323 st.subheader('Project Keywords')
324 st.json(keywords)
325
326 # ... Streamlit UI ...
327 st.subheader('Public Repositories')
328 st.json([repo['full_name'] for repo in repos])
329
330 # ... Streamlit UI ...
331 st.subheader('Project Keywords')
332 st.json(keywords)
333
334 # ... Streamlit UI ...
335 st.subheader('Public Repositories')
336 st.json([repo['full_name'] for repo in repos])
337
338 # ... Streamlit UI ...
339 st.subheader('Project Keywords')
340 st.json(keywords)
341
342 # ... Streamlit UI ...
343 st.subheader('Public Repositories')
344 st.json([repo['full_name'] for repo in repos])
345
346 # ... Streamlit UI ...
347 st.subheader('Project Keywords')
348 st.json(keywords)
349
350 # ... Streamlit UI ...
351 st.subheader('Public Repositories')
352 st.json([repo['full_name'] for repo in repos])
353
354 # ... Streamlit UI ...
355 st.subheader('Project Keywords')
356 st.json(keywords)
357
358 # ... Streamlit UI ...
359 st.subheader('Public Repositories')
360 st.json([repo['full_name'] for repo in repos])
361
362 # ... Streamlit UI ...
363 st.subheader('Project Keywords')
364 st.json(keywords)
365
366 # ... Streamlit UI ...
367 st.subheader('Public Repositories')
368 st.json([repo['full_name'] for repo in repos])
369
370 # ... Streamlit UI ...
371 st.subheader('Project Keywords')
372 st.json(keywords)
373
374 # ... Streamlit UI ...
375 st.subheader('Public Repositories')
376 st.json([repo['full_name'] for repo in repos])
377
378 # ... Streamlit UI ...
379 st.subheader('Project Keywords')
380 st.json(keywords)
381
382 # ... Streamlit UI ...
383 st.subheader('Public Repositories')
384 st.json([repo['full_name'] for repo in repos])
385
386 # ... Streamlit UI ...
387 st.subheader('Project Keywords')
388 st.json(keywords)
389
390 # ... Streamlit UI ...
391 st.subheader('Public Repositories')
392 st.json([repo['full_name'] for repo in repos])
393
394 # ... Streamlit UI ...
395 st.subheader('Project Keywords')
396 st.json(keywords)
397
398 # ... Streamlit UI ...
399 st.subheader('Public Repositories')
400 st.json([repo['full_name'] for repo in repos])
401
402 # ... Streamlit UI ...
403 st.subheader('Project Keywords')
404 st.json(keywords)
405
406 # ... Streamlit UI ...
407 st.subheader('Public Repositories')
408 st.json([repo['full_name'] for repo in repos])
409
410 # ... Streamlit UI ...
411 st.subheader('Project Keywords')
412 st.json(keywords)
413
414 # ... Streamlit UI ...
415 st.subheader('Public Repositories')
416 st.json([repo['full_name'] for repo in repos])
417
418 # ... Streamlit UI ...
419 st.subheader('Project Keywords')
420 st.json(keywords)
421
422 # ... Streamlit UI ...
423 st.subheader('Public Repositories')
424 st.json([repo['full_name'] for repo in repos])
425
426 # ... Streamlit UI ...
427 st.subheader('Project Keywords')
428 st.json(keywords)
429
430 # ... Streamlit UI ...
431 st.subheader('Public Repositories')
432 st.json([repo['full_name'] for repo in repos])
433
434 # ... Streamlit UI ...
435 st.subheader('Project Keywords')
436 st.json(keywords)
437
438 # ... Streamlit UI ...
439 st.subheader('Public Repositories')
440 st.json([repo['full_name'] for repo in repos])
441
442 # ... Streamlit UI ...
443 st.subheader('Project Keywords')
444 st.json(keywords)
445
446 # ... Streamlit UI ...
447 st.subheader('Public Repositories')
448 st.json([repo['full_name'] for repo in repos])
449
450 # ... Streamlit UI ...
451 st.subheader('Project Keywords')
452 st.json(keywords)
453
454 # ... Streamlit UI ...
455 st.subheader('Public Repositories')
456 st.json([repo['full_name'] for repo in repos])
457
458 # ... Streamlit UI ...
459 st.subheader('Project Keywords')
460 st.json(keywords)
461
462 # ... Streamlit UI ...
463 st.subheader('Public Repositories')
464 st.json([repo['full_name'] for repo in repos])
465
466 # ... Streamlit UI ...
467 st.subheader('Project Keywords')
468 st.json(keywords)
469
470 # ... Streamlit UI ...
471 st.subheader('Public Repositories')
472 st.json([repo['full_name'] for repo in repos])
473
474 # ... Streamlit UI ...
475 st.subheader('Project Keywords')
476 st.json(keywords)
477
478 # ... Streamlit UI ...
479 st.subheader('Public Repositories')
480 st.json([repo['full_name'] for repo in repos])
481
482 # ... Streamlit UI ...
483 st.subheader('Project Keywords')
484 st.json(keywords)
485
486 # ... Streamlit UI ...
487 st.subheader('Public Repositories')
488 st.json([repo['full_name'] for repo in repos])
489
490 # ... Streamlit UI ...
491 st.subheader('Project Keywords')
492 st.json(keywords)
493
494 # ... Streamlit UI ...
495 st.subheader('Public Repositories')
496 st.json([repo['full_name'] for repo in repos])
497
498 # ... Streamlit UI ...
499 st.subheader('Project Keywords')
500 st.json(keywords)
501
502 # ... Streamlit UI ...
503 st.subheader('Public Repositories')
504 st.json([repo['full_name'] for repo in repos])
505
506 # ... Streamlit UI ...
507 st.subheader('Project Keywords')
508 st.json(keywords)
509
510 # ... Streamlit UI ...
511 st.subheader('Public Repositories')
512 st.json([repo['full_name'] for repo in repos])
513
514 # ... Streamlit UI ...
515 st.subheader('Project Keywords')
516 st.json(keywords)
517
518 # ... Streamlit UI ...
519 st.subheader('Public Repositories')
520 st.json([repo['full_name'] for repo in repos])
521
522 # ... Streamlit UI ...
523 st.subheader('Project Keywords')
524 st.json(keywords)
525
526 # ... Streamlit UI ...
527 st.subheader('Public Repositories')
528 st.json([repo['full_name'] for repo in repos])
529
530 # ... Streamlit UI ...
531 st.subheader('Project Keywords')
532 st.json(keywords)
533
534 # ... Streamlit UI ...
535 st.subheader('Public Repositories')
536 st.json([repo['full_name'] for repo in repos])
537
538 # ... Streamlit UI ...
539 st.subheader('Project Keywords')
540 st.json(keywords)
541
542 # ... Streamlit UI ...
543 st.subheader('Public Repositories')
544 st.json([repo['full_name'] for repo in repos])
545
546 # ... Streamlit UI ...
547 st.subheader('Project Keywords')
548 st.json(keywords)
549
550 # ... Streamlit UI ...
551 st.subheader('Public Repositories')
552 st.json([repo['full_name'] for repo in repos])
553
554 # ... Streamlit UI ...
555 st.subheader('Project Keywords')
556 st.json(keywords)
557
558 # ... Streamlit UI ...
559 st.subheader('Public Repositories')
560 st.json([repo['full_name'] for repo in repos])
561
562 # ... Streamlit UI ...
563 st.subheader('Project Keywords')
564 st.json(keywords)
565
566 # ... Streamlit UI ...
567 st.subheader('Public Repositories')
568 st.json([repo['full_name'] for repo in repos])
569
570 # ... Streamlit UI ...
571 st.subheader('Project Keywords')
572 st.json(keywords)
573
574 # ... Streamlit UI ...
575 st.subheader('Public Repositories')
576 st.json([repo['full_name'] for repo in repos])
577
578 # ... Streamlit UI ...
579 st.subheader('Project Keywords')
580 st.json(keywords)
581
582 # ... Streamlit UI ...
583 st.subheader('Public Repositories')
584 st.json([repo['full_name'] for repo in repos])
585
586 # ... Streamlit UI ...
587 st.subheader('Project Keywords')
588 st.json(keywords)
589
590 # ... Streamlit UI ...
591 st.subheader('Public Repositories')
592 st.json([repo['full_name'] for repo in repos])
593
594 # ... Streamlit UI ...
595 st.subheader('Project Keywords')
596 st.json(keywords)
597
598 # ... Streamlit UI ...
599 st.subheader('Public Repositories')
600 st.json([repo['full_name'] for repo in repos])
601
602 # ... Streamlit UI ...
603 st.subheader('Project Keywords')
604 st.json(keywords)
605
606 # ... Streamlit UI ...
607 st.subheader('Public Repositories')
608 st.json([repo['full_name'] for repo in repos])
609
610 # ... Streamlit UI ...
611 st.subheader('Project Keywords')
612 st.json(keywords)
613
614 # ... Streamlit UI ...
615 st.subheader('Public Repositories')
616 st.json([repo['full_name'] for repo in repos])
617
618 # ... Streamlit UI ...
619 st.subheader('Project Keywords')
620 st.json(keywords)
621
622 # ... Streamlit UI ...
623 st.subheader('Public Repositories')
624 st.json([repo['full_name'] for repo in repos])
625
626 # ... Streamlit UI ...
627 st.subheader('Project Keywords')
628 st.json(keywords)
629
630 # ... Streamlit UI ...
631 st.subheader('Public Repositories')
632 st.json([repo['full_name'] for repo in repos])
633
634 # ... Streamlit UI ...
635 st.subheader('Project Keywords')
636 st.json(keywords)
637
638 # ... Streamlit UI ...
639 st.subheader('Public Repositories')
640 st.json([repo['full_name'] for repo in repos])
641
642 # ... Streamlit UI ...
643 st.subheader('Project Keywords')
644 st.json(keywords)
645
646 # ... Streamlit UI ...
647 st.subheader('Public Repositories')
648 st.json([repo['full_name'] for repo in repos])
649
650 # ... Streamlit UI ...
651 st.subheader('Project Keywords')
652 st.json(keywords)
653
654 # ... Streamlit UI ...
655 st.subheader('Public Repositories')
656 st.json([repo['full_name'] for repo in repos])
657
658 # ... Streamlit UI ...
659 st.subheader('Project Keywords')
660 st.json(keywords)
661
662 # ... Streamlit UI ...
663 st.subheader('Public Repositories')
664 st.json([repo['full_name'] for repo in repos])
665
666 # ... Streamlit UI ...
667 st.subheader('Project Keywords')
668 st.json(keywords)
669
670 # ... Streamlit UI ...
671 st.subheader('Public Repositories')
672 st.json([repo['full_name'] for repo in repos])
673
674 # ... Streamlit UI ...
675 st.subheader('Project Keywords')
676 st.json(keywords)
677
678 # ... Streamlit UI ...
679 st.subheader('Public Repositories')
680 st.json([repo['full_name'] for repo in repos])
681
682 # ... Streamlit UI ...
683 st.subheader('Project Keywords')
684 st.json(keywords)
685
686 # ... Streamlit UI ...
687 st.subheader('Public Repositories')
688 st.json([repo['full_name'] for repo in repos])
689
690 # ... Streamlit UI ...
691 st.subheader('Project Keywords')
692 st.json(keywords)
693
694 # ... Streamlit UI ...
695 st.subheader('Public Repositories')
696 st.json([repo['full_name'] for repo in repos])
697
698 # ... Streamlit UI ...
699 st.subheader('Project Keywords')
700 st.json(keywords)
701
702 # ... Streamlit UI ...
703 st.subheader('Public Repositories')
704 st.json([repo['full_name'] for repo in repos])
705
706 # ... Streamlit UI ...
707 st.subheader('Project Keywords')
708 st.json(keywords)
709
710 # ... Streamlit UI ...
711 st.subheader('Public Repositories')
712 st.json([repo['full_name'] for repo in repos])
713
714 # ... Streamlit UI ...
715 st.subheader('Project Keywords')
716 st.json(keywords)
717
718 # ... Streamlit UI ...
719 st.subheader('Public Repositories')
720 st.json([repo['full_name'] for repo in repos])
721
722 # ... Streamlit UI ...
723 st.subheader('Project Keywords')
724 st.json(keywords)
725
726 # ... Streamlit UI ...
727 st.subheader('Public Repositories')
728 st.json([repo['full_name'] for repo in repos])
729
730 # ... Streamlit UI ...
731 st.subheader('Project Keywords')
732 st.json(keywords)
733
734 # ... Streamlit UI ...
735 st.subheader('Public Repositories')
736 st.json([repo['full_name'] for repo in repos])
737
738 # ... Streamlit UI ...
739 st.subheader('Project Keywords')
740 st.json(keywords)
741
742 # ... Streamlit UI ...
743 st.subheader('Public Repositories')
744 st.json([repo['full_name'] for repo in repos])
745
746 # ... Streamlit UI ...
747 st.subheader('Project Keywords')
748 st.json(keywords)
749
750 # ... Streamlit UI ...
751 st.subheader('Public Repositories')
752 st.json([repo['full_name'] for repo in repos])
753
754 # ... Streamlit UI ...
755 st.subheader('Project Keywords')
756 st.json(keywords)
757
758 # ... Streamlit UI ...
759 st.subheader('Public Repositories')
760 st.json([repo['full_name'] for repo in repos])
761
762 # ... Streamlit UI ...
763 st.subheader('Project Keywords')
764 st.json(keywords)
765
766 # ... Streamlit UI ...
767 st.subheader('Public Repositories')
768 st.json([repo['full_name'] for repo in repos])
769
770 # ... Streamlit UI ...
771 st.subheader('Project Keywords')
772 st.json(keywords)
773
774 # ... Streamlit UI ...
775 st.subheader('Public Repositories')
776 st.json([repo['full_name'] for repo in repos])
777
778 # ... Streamlit UI ...
779 st.subheader('Project Keywords')
780 st.json(keywords)
781
782 # ... Streamlit UI ...
783 st.subheader('Public Repositories')
784 st.json([repo['full_name'] for repo in repos])
785
786 # ... Streamlit UI ...
787 st.subheader('Project Keywords')
788 st.json(keywords)
789
790 # ... Streamlit UI ...
791 st.subheader('Public Repositories')
792 st.json([repo['full_name'] for repo in repos])
793
794 # ... Streamlit UI ...
795 st.subheader('Project Keywords')
796 st.json(keywords)
797
798 # ... Streamlit UI ...
799 st.subheader('Public Repositories')
800 st.json([repo['full_name'] for repo in repos])
801
802 # ... Streamlit UI ...
803 st.subheader('Project Keywords')
804 st.json(keywords)
805
806 # ... Streamlit UI ...
807 st.subheader('Public Repositories')
808 st.json([repo['full_name'] for repo in repos])
809
810 # ... Streamlit UI ...
811 st.subheader('Project Keywords')
812 st.json(keywords)
813
814 # ... Streamlit UI ...
815 st.subheader('Public Repositories')
816 st.json([repo['full_name'] for repo in repos])
817
818 # ... Streamlit UI ...
819 st.subheader('Project Keywords')
820 st.json(keywords)
821
822 # ... Streamlit UI ...
823 st.subheader('Public Repositories')
824 st.json([repo['full_name'] for repo in repos])
825
826 # ... Streamlit UI ...
827 st.subheader('Project Keywords')
828 st.json(keywords)
829
830 # ... Streamlit UI ...
831 st.subheader('Public Repositories')
832 st.json([repo['full_name'] for repo in repos])
833
834 # ... Streamlit UI ...
835 st.subheader('Project Keywords')
836 st.json(keywords)
837
838 # ... Streamlit UI ...
839 st.subheader('Public Repositories')
840 st.json([repo['full_name'] for repo in repos])
841
842 # ... Streamlit UI ...
843 st.subheader('Project Keywords')
844 st.json(keywords)
845
846 # ... Streamlit UI ...
847 st.subheader('Public Repositories')
848 st.json([repo['full_name'] for repo in repos])
849
850 # ... Streamlit UI ...
851 st.subheader('Project Keywords')
852 st.json(keywords)
853
854 # ... Streamlit UI ...
855 st.subheader('Public Repositories')
856 st.json([repo['full_name'] for repo in repos])
857
858 # ... Streamlit UI ...
859 st.subheader('Project Keywords')
860 st.json(keywords)
861
862 # ... Streamlit UI ...
863 st.subheader('Public Repositories')
864 st.json([repo['full_name'] for repo in repos])
865
866 # ... Streamlit UI ...
867 st.subheader('Project Keywords')
868 st.json(keywords)
869
870 # ... Streamlit UI ...
871 st.subheader('Public Repositories')
872 st.json([repo['full_name'] for repo in repos])
873
874 # ... Streamlit UI ...
875 st.subheader('Project Keywords')
876 st.json(keywords)
877
878 # ... Streamlit UI ...
879 st.subheader('Public Repositories')
880 st.json([repo['full_name'] for repo in repos])
881
882 # ... Streamlit UI ...
883 st.subheader('Project Keywords')
884 st.json(keywords)
885
886 # ... Streamlit UI ...
887 st.subheader('Public Repositories')
888 st.json([repo['full_name'] for repo in repos])
889
890 # ... Streamlit UI ...
891 st.subheader('Project Keywords')
892 st.json(keywords)
893
894 # ... Streamlit UI ...
895 st.subheader('Public Repositories')
896 st.json([repo['full_name'] for repo in repos])
897
898 # ... Streamlit UI ...
899 st.subheader('Project Keywords')
900 st.json(keywords)
901
902 # ... Streamlit UI ...
903 st.subheader('Public Repositories')
904 st.json([repo['full_name'] for repo in repos])
905
906 # ... Streamlit UI ...
907 st.subheader('Project Keywords')
908 st.json(keywords)
909
910 # ... Streamlit UI ...
911 st.subheader('Public Repositories')
912 st.json([repo['full_name'] for repo in repos])
913
914 # ... Streamlit UI ...
915 st.subheader('Project Keywords')
916 st.json(keywords)
917
918 # ... Streamlit UI ...
919 st.subheader('Public Repositories')
920 st.json([repo['full_name'] for repo in repos])
921
922 # ... Streamlit UI ...
923 st.subheader('Project Keywords')
924 st.json(keywords)
925
926 # ... Streamlit UI ...
927 st.subheader('Public Repositories')
928 st.json([repo['full_name'] for repo in repos])
929
930 # ... Streamlit UI ...
931 st.subheader('Project Keywords')
932 st.json(keywords)
933
934 # ... Streamlit UI ...
935 st.subheader('Public Repositories')
936 st.json([repo['full_name'] for repo in repos])
937
938 # ... Streamlit UI ...
939 st.subheader('Project Keywords')
940 st.json(keywords)
941
942 # ... Streamlit UI ...
943 st.subheader('Public Repositories')
944 st.json([repo['full_name'] for repo in repos])
945
946 # ... Streamlit UI ...
947 st.subheader('Project Keywords')
948 st.json(keywords)
949
950 # ... Streamlit UI ...
951 st.subheader('Public Repositories')
952 st.json([repo['full_name'] for repo in repos])
953
954 # ... Streamlit UI ...
955 st.subheader('Project Keywords')
956 st.json(keywords)
957
958 # ... Streamlit UI ...
959 st.subheader('Public Repositories')
960 st.json([repo['full_name'] for repo in repos])
961
962 # ... Streamlit UI ...
963 st.subheader('Project Keywords')
964 st.json(keywords)
965
966 # ... Streamlit UI ...
967 st.subheader('Public Repositories')
968 st.json([repo['full_name'] for repo in repos])
969
970 # ... Streamlit UI ...
971 st.subheader('Project Keywords')
972 st.json(keywords)
973
974 # ... Streamlit UI ...
975 st.subheader('Public Repositories')
976 st.json([repo['full_name'] for repo in repos])
977
978 # ... Streamlit UI ...
979 st.subheader('Project Keywords')
980 st.json(keywords)
981
982 # ... Streamlit UI ...
983 st.subheader('Public Repositories')
984 st.json([repo['full_name'] for repo in repos])
985
986 # ... Streamlit UI ...
987 st.subheader('Project Keywords')
988 st.json(keywords)
989
990 # ... Streamlit UI ...
991 st.subheader('Public Repositories')
992 st.json([repo['full_name'] for repo in repos])
993
994 # ... Streamlit UI ...
995 st.subheader('Project Keywords')
996 st.json(keywords)
997
998 # ... Streamlit UI ...
999 st.subheader('Public Repositories')
1000 st.json([repo['full_name'] for repo in repos])
1001
1002 # ... Streamlit UI ...
1003 st.subheader('Project Keywords')
1004 st.json(keywords)
1005
1006 # ... Streamlit UI ...
1007 st.subheader('Public Repositories')
1008 st.json([repo['full_name'] for repo in repos])
1009
1010 # ... Streamlit UI ...
1011 st.subheader('Project Keywords')
1012 st.json(keywords)
1013
1014 # ... Streamlit UI ...
1015 st.subheader('Public Repositories')
1016 st.json([repo['full_name'] for repo in repos])
1017
1018 # ... Streamlit UI ...
1019 st.subheader('Project Keywords')
1020 st.json(keywords)
1021
1022 # ... Streamlit UI ...
1023 st.subheader('Public Repositories')
1024 st.json([repo['full_name'] for repo in repos])
1025
1026 # ... Streamlit UI ...
1027 st.subheader('Project Keywords')
1028 st.json(keywords)
1029
1030 # ... Streamlit UI ...
1031 st.subheader('Public Repositories')
1032 st.json([repo['full_name'] for repo in repos])
1033
1034 # ... Streamlit UI ...
1035 st.subheader('Project Keywords')
1036 st.json(keywords)
1037
1038 # ... Streamlit UI ...
1039 st.subheader('Public Repositories')
1040 st.json([repo['full_name'] for repo in repos])
1041
1042 # ... Streamlit UI ...
1043 st.subheader('Project Keywords')
1044 st.json(keywords)
1045
1046 # ... Streamlit UI ...
1047 st.subheader('Public Repositories')
1048 st.json([repo['full_name'] for repo in repos])
1049
1050 # ... Streamlit UI ...
1051 st.subheader('Project Keywords')
1052 st.json(keywords)
1053
1054 # ... Streamlit UI ...
1055 st.subheader('Public Repositories')
1056 st.json([repo['full_name'] for repo in repos])
1057
1058 # ... Streamlit UI ...
1059 st.subheader('Project Keywords')
1060 st.json(keywords)
1061
1062 # ... Streamlit UI ...
1063 st.subheader('Public Repositories')
1064 st.json([repo['full_name'] for repo in repos])
1065
1066 # ... Streamlit UI ...
1067 st.subheader('Project Keywords')
1068 st.json(keywords)
1069
1070 # ... Streamlit UI ...
1071 st.subheader('Public Repositories')
1072 st.json([repo['full_name'] for repo in repos])
1073
1074 # ... Streamlit UI ...
1075 st.subheader('Project Keywords')
1076 st.json(keywords)
1077
1078 # ... Streamlit UI ...
1079 st.subheader('Public Repositories')
1080 st.json([repo['full_name'] for repo in repos])
1081
1082 # ... Streamlit UI ...
1083 st.subheader('Project Keywords')
1084 st.json(keywords)
1085
1086 # ... Streamlit UI ...
1087 st.subheader('Public Repositories')
1088 st.json([repo['full_name'] for repo in repos])
1089
1090 # ... Streamlit UI ...
1091 st.subheader('Project Keywords')
1092 st.json(keywords)
1093
1094 # ... Streamlit UI ...
1095 st.subheader('Public Repositories')
1096 st.json([repo['full_name'] for repo in repos])
1097
1098 # ... Streamlit UI ...
1099 st.subheader('Project Keywords')
1100 st.json(keywords)
1101
1102 # ... Streamlit UI ...
1103 st.subheader('Public Repositories')
1104 st.json([repo['full_name'] for repo in repos])
1105
1106 # ... Streamlit UI ...
1107 st.subheader('Project Keywords')
1108 st.json(keywords)
1109
1110 # ... Streamlit UI ...
1111 st.subheader('Public Repositories')
1112 st.json([repo['full_name'] for repo in repos])
1113
1114 # ... Streamlit UI ...
1115 st.subheader('Project Keywords')
1116 st.json(keywords)
1117
1118 # ... Streamlit UI ...
1119 st.subheader('Public Repositories')
1120 st.json([repo['full_name'] for repo in repos])
1121
1122 # ... Streamlit UI ...
1123 st.subheader('Project Keywords')
1124 st.json(keywords)
1125
1126 # ... Streamlit UI ...
1127 st.subheader('Public Repositories')
1128 st.json([repo['full_name'] for repo in repos])
1129
1130 # ... Streamlit UI ...
1131 st.subheader('Project Keywords')
1132 st.json(keywords)
1133
1134 # ... Streamlit UI ...
1135 st.subheader('Public Repositories')
1136 st.json([repo['full_name'] for repo in repos])
1137
1138 # ... Streamlit UI ...
1139 st.subheader('Project Keywords')
1140 st.json(keywords)
1141
1142 # ... Streamlit UI ...
1143 st.subheader('Public Repositories')
1144 st.json([repo['full_name'] for repo in repos])
1145
1146 # ... Streamlit UI ...
1147 st.subheader('Project Keywords')
1148 st.json(keywords)
1149
1150 # ... Streamlit UI ...
1151 st.subheader('Public Repositories')
1152 st.json([repo['full_name'] for repo in repos])
1153
1154 # ... Streamlit UI ...
1155 st.subheader('Project Keywords')
1156 st.json(keywords)
1157
1158 # ... Streamlit UI ...
1159 st.subheader('Public Repositories')
1160 st.json([repo['full_name'] for repo in repos])
1161
1162 # ... Streamlit UI ...
1163 st.subheader('Project Keywords')
1164 st.json(keywords)
1165
1166 # ... Streamlit UI ...
1167 st.subheader('Public Repositories')
1168 st.json([repo['full_name'] for repo in repos])
1169
1170 # ... Streamlit UI ...
1171 st.subheader('Project Keywords')
1172 st.json(keywords)
1173
1174 # ... Streamlit UI ...
1175 st.subheader('Public Repositories')
1176 st.json([repo['full_name'] for repo in repos])
1177
1178 # ... Streamlit UI ...
1179 st.subheader('Project Keywords')
1180 st.json(keywords)
1181
1182 # ... Streamlit UI ...
1183 st.subheader('Public Repositories')
1184 st.json([repo['full_name'] for repo in repos])
1185
1186 # ... Streamlit UI ...
1187 st.subheader('Project Keywords')
1188 st.json(keywords)
1189
1190 # ... Streamlit UI ...
1191 st.subheader('Public Repositories')
1192 st.json([repo['full_name'] for repo in repos])
1193
1194 # ... Streamlit UI ...
1195 st.subheader('Project Keywords')
1196 st.json(keywords)
1197
1198 # ... Streamlit UI ...
1199 st.subheader('Public Repositories')
1200 st.json([repo['full_name'] for repo in repos])
1201
1202 # ... Streamlit UI ...
1203 st.subheader('Project Keywords')
1204 st.json(keywords)
1205
1206 # ... Streamlit UI ...
1207 st.subheader('Public Repositories')
1208 st.json([repo['full_name'] for repo in repos])
1209
1210 # ... Streamlit UI ...
1211 st.subheader('Project Keywords')
1212 st.json(keywords)
1213
1214 # ... Streamlit UI ...
1215 st.subheader('Public Repositories')
1216 st.json([repo['full_name'] for repo in repos])
1217
1218 # ... Streamlit UI ...
```