Fall 2021 Advanced Software Engineering
T2: Revised Project Proposal

Team Akea

Name	UNI	Github Username
Eric Fan	xf2218	EricFan24
Adit V Deshmukh	avd2133	adit-10
Alexandra Cheng	yc3492	AlexandraCheng
Kashish Chanana	kc3419	KashishChanana

Part 1: Meeting with IA

We met with our IA mentor, Kun, on Friday, Oct. 22nd.

Part 2: Write a few paragraphs that provide an overview of the service that your team would like to develop and answer these three sets of questions:

1. What will your service do? What kind of functionality or features will it provide?

We propose to build a service called "Smart Bookmarks". More often than not, we forget to properly designate bookmarks to folders or find it difficult to scavenge through our long list of bookmarks to find the one we've been looking for. We believe associating bookmarks with certain defining words can help to search & sort the bookmarks better.

For this project we will focus on news articles for initial implementation and testing. We plan to expand the scope of acceptable urls to other categories of websites.

As an example, if I were to bookmark the site https://www.nytimes.com/interactive/2021/us/covid-cases.html, I can use words/tags such as 'Covid', 'New York', '2021' to better categorize the website and make the searching experience for bookmarks easier, quicker, and more intuitive.

Features & Functionality:

1. Our service will take in the URL of the website we want to bookmark and automatically generate tags for the website using natural language processing. Therefore, each website URL will be associated with tags that define the content of the website.
2. The service will also allow the user to modify these tags or add their own.

3. The user can send these tags as a query parameter for the service and a list of URLs conforming to the query tag list will be returned back to the user.
4. The user can also search through the bookmark list based on time (“today”, “yesterday’’, “last week”, “last month”, “last 3 months”, etc.), keywords extracted from titles/metadata of the web pages, as well as based on the genre of the content (‘news’, ‘technology’, ‘sci-fi’ etc.)

Data Persistence:
Yes, we’ll make use of a *Postgresql* database for persisting data related to clientID, urls, and tags. This data can be accessed cross-executions.
Multi-Client Support:
The service will support multiple clients. The clients will be distinguished using Client ID and password. For testing, we aim to use the Authorization header of Postman and use OAuth 2.0 to generate access tokens or client secrets. Having specific IDs and secrets allows us to protect client data from other clients as the bookmarks pertaining to the client in consideration will only be returned to him/her.

Natural Language Processing for tag generation -

1. Extract metadata, headline and first paragraph using BeautifulSoup
2. Remove stop words using Gensim/NLTK
3. Do lemmatization and extract keywords using NLTK/Gensim and pretrained neural networks
4. Do clustering based on extracted keywords (using TF-IDF, for example)
5. If more than 10, keep a maximum of 10 keywords (based on frequency/semantic similarities)

Our service supports multiple different applications and not just the backend of a single app. Most web browser applications like Google Chrome, Mozilla Firefox, Safari can utilize our service to allow Smart Bookmarking. Further, any user/service that requires auto-tag generation from the contents of web pages can use our service as well.

Core Endpoints:

Log in/out and authenticate with credentials
Add URL for automatic parsing
Add URL and corresponding keywords
Query URL, return corresponding keywords
Query URL, return list of similar URLs (based on clustering of keywords)
Query keyword(s), return corresponding URLs
Add/delete keywords from existing URLs

2. **Who or what will be its users? What might they use the functionality for?**

   a. As an online reader (indirect user), the user wants to save multiple relevant articles so that they can retrieve them later.
   b. As a researcher or journalist (indirect user), the user wants to query keywords/categories from saved sources so that they won't need to read them again one by one.
   c. As an app/browser (direct user) managing bookmarks for multiple users, the app developer wants to read/write multiple accounts at the same time, so that they can manage the accounts and bookmarks for all of their users.

Usefulness:
Our service supports multiple different applications and not just the backend of a single app. Most web browser applications like Google Chrome, Mozilla Firefox, Safari can utilize our service to allow Smart Bookmarking. Further, any user/service that requires auto-tag generation from the contents of web pages can use our service as well.

3. What kind of data will your service create or accumulate? What will the data be used for?

The service will use Natural Language Processing to extract keywords and generate tags associated with URLs bookmarked by the users. For each user, the service will maintain information about the bookmarks, tags associated with the bookmarks, and metadata such as the category of URL, timestamp, etc. This information will be stored in a relational database.

For generating the tags, the service will retrieve information from the link provided and run the text through a tag generation model.

Apart from this, the service will also maintain information about users and their authentication information.

Part 3: Write a few paragraphs that describe, at a high level, how you plan to test the functionality of your service without any clients, GUI, or otherwise.

1. How will you test that your service does what it is supposed to do and provides the intended functionality?

We will be using Postman to test the API calls. When clients want to create a new bookmark or get a collection of bookmarks with a keyword, we will use the GET method to retrieve the data (either from our database or a webpage), so Postman GET calls can be used to test the following:

- a 200 status code is returned (valid request).
- correct data is returned (in this case, the webpage URL or the bookmark collection associated with the keyword)

Similarly, we use the POST/PUT method to add the bookmark URLs to the database. We can test this by manually creating a POST/PUT request in Postman, checking the returned status code (200), and using the GET method to check the updated database (information is stored).

We will test using news articles with pre-existing categories, such as sports, education, healthcare, food, etc. To test our automatic keyword extraction, we can manually compare the tags generated by our function with existing categories as well as the the content on the page, and see how connected they are.

To make sure that clients can only access their own bookmarks, we plan to use the Authorization header of Postman and use OAuth 2.0 to generate access tokens or client secrets (different client IDs). If they try to get a bookmark from other clients, we deny the access and return an error status code.

2. How will you check that your service does not behave badly if its clients use it in unintended ways or provide invalid inputs?

When clients try to use the service in unintended ways, we will stop processing the requests and return status codes like 400 (Bad Request) or 405 (Method Not Allowed). If the input is invalid, the service will return 400 (Bad Request). We can test this by using Postman to make an invalid request and then check the returned status code as well as the user database to make sure the unintended usage does not update anything in the database.

3. How will you test that your service handles its data the way it's supposed to?

For all the data we receive (e.g. page URLs, user-defined tags), we will create a data entry or update an existing entry inside the database. To make sure the complete data is stored/updated in the correct format and position, we can create some test data and insert them into the database with a Postman POST request, shut down the server, then restart the server and retrieve the data with a GET request. If the retrieved data matches the inserted data, then we know that our service handles user data correctly.

Part 4: List of libraries, frameworks, tools, etc. we will potentially use:

- *Auth0* for authentication: https://auth0.com/docs/
- *Postman* for testing GET/POST calls: https://www.postman.com/
- *BeautifulSoup* for data extraction from HTML files: https://www.crummy.com/software/BeautifulSoup/bs4/doc/
- Gensim, sklearn, NLTK, Tensorflow for NLP
- *Postgresql* for database: https://www.postgresql.org/
- Google Python Style Guide: https://google.github.io/styleguide/pyguide.html
- Bug finder: *flake8*
- Test runner: *unittest*
- Coverage tracker: *coverage.py*