# The Feedback Analyzer
User guide

Concepts, Standard of Procedure, Future suggestions

*Let us briefly recap:*

# The Problem & Solution

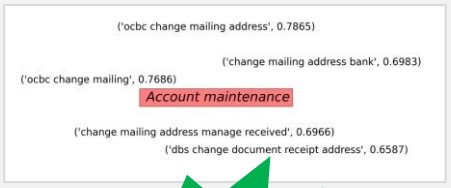
| Net E-B Feedback | Sub-Category | Category |
|---|---|---|
| Instead of charging GIRO charges for each transactions, combined all GIRO charges and charge as 1 transaction by end of each month, like you do for cheque transactions. | I want transfer charges to be consolidated and shown monthly | Transfer charges |

1-2 days

**The Problem**: How can we improve efficiency when gathering insights so that customer verbatims can be easily translated into product features?

| Net E-B Feedback |
|---|
| Instead of charging GIRO charges for each transactions, combined all GIRO charges and charge as 1 transaction by end of each month, like you do for cheque transactions. |

| Category |
|---|
| Transfer charges |




1-2 hours

**The Solution**: Automating the feedback analysis using AI to categorize the feedback and generate key-phrases per category

# User manual

Automated Feedback Analayzer

Step1: Uploading feedback in CSV

⚠ Before uploading, make sure you have prepared the following:

- Place your feedback under a column and name the column header as: 'Feedback'
- Create another column on the right and name the column header as: 'Category'
- Save your file as CSV-UTF8 (Comma Delimited) before uploading

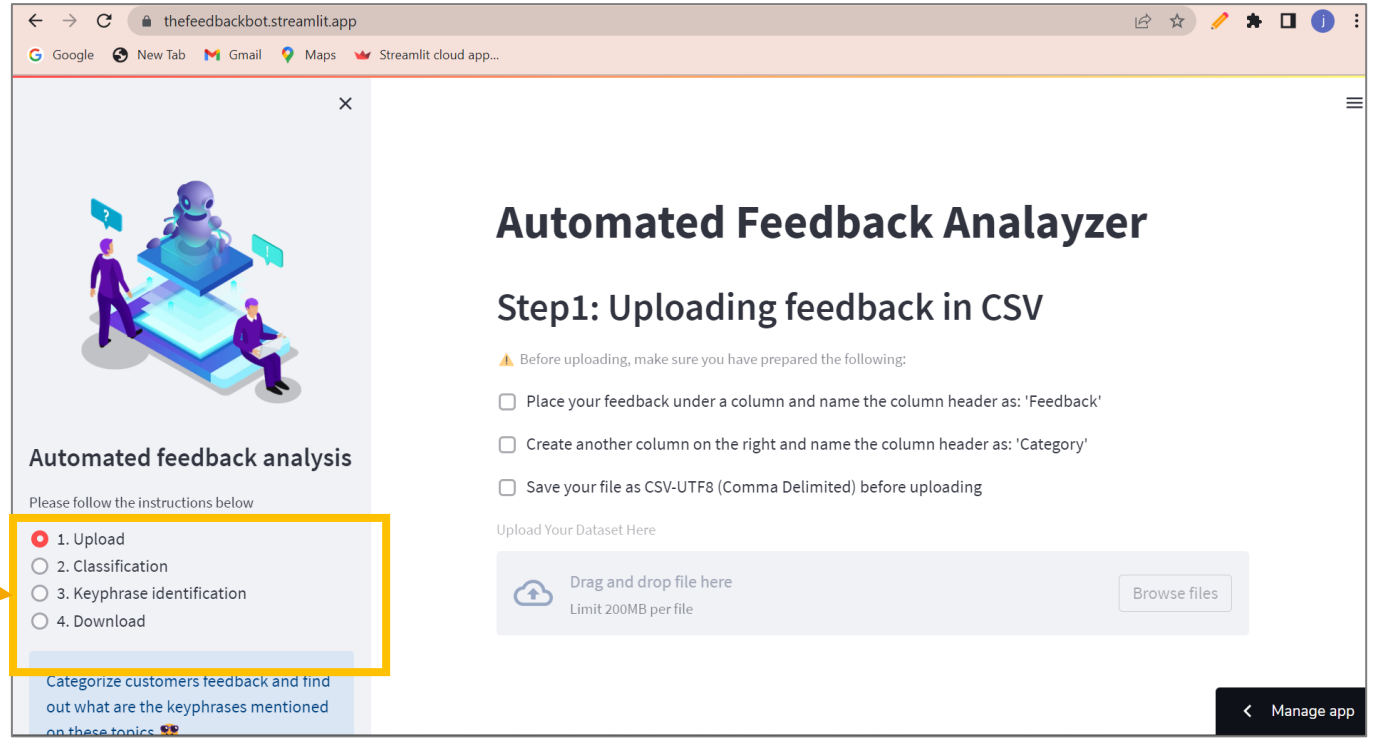
Upload Your Dataset Here

Drag and drop file here

Limit 200MB per file

# Landing page

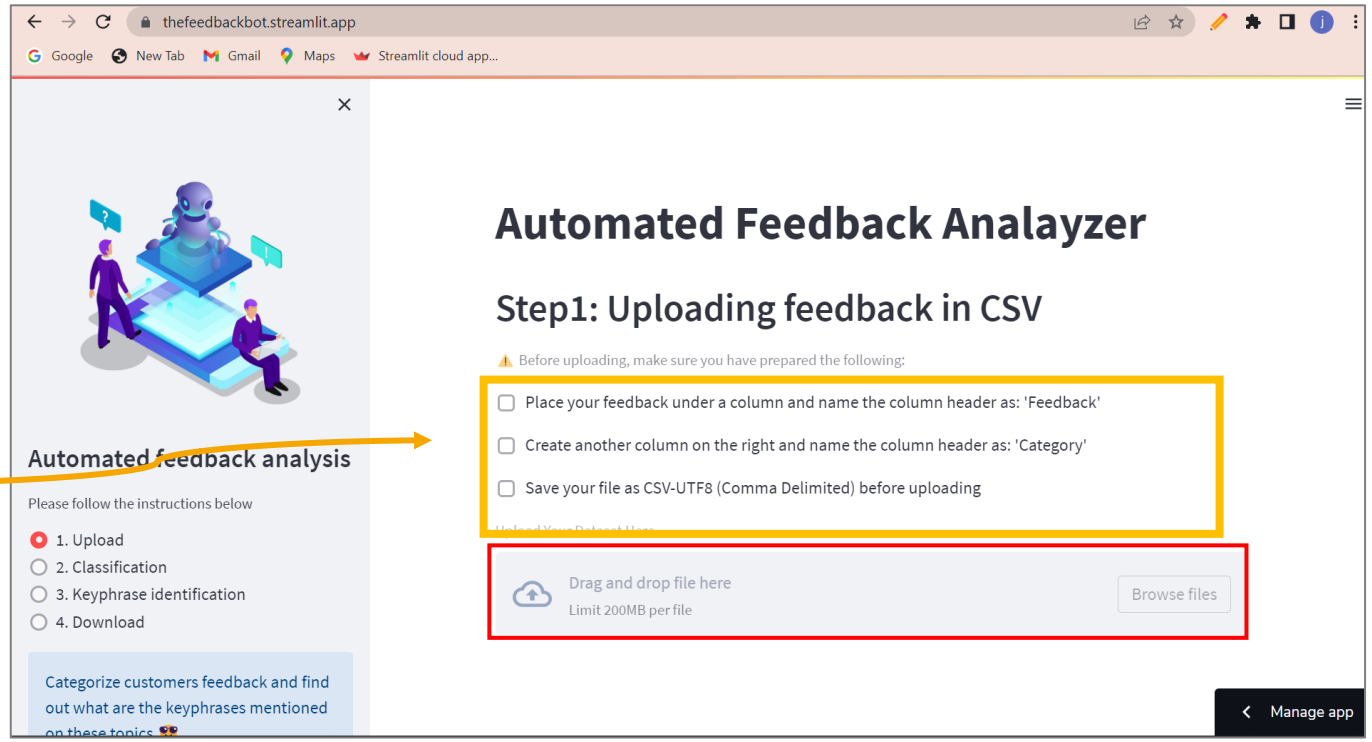
- To access the public web app, visit: https://thefeedbackbot.streamlit.app/
- To start analyzing the feedback, follow the instructions on the navigation bar on the left
- There are 4 steps to analyzing the feedback:
    1. Uploading file
    2. Categorizing
    3. Generate key phrase per category
    4. Downloading

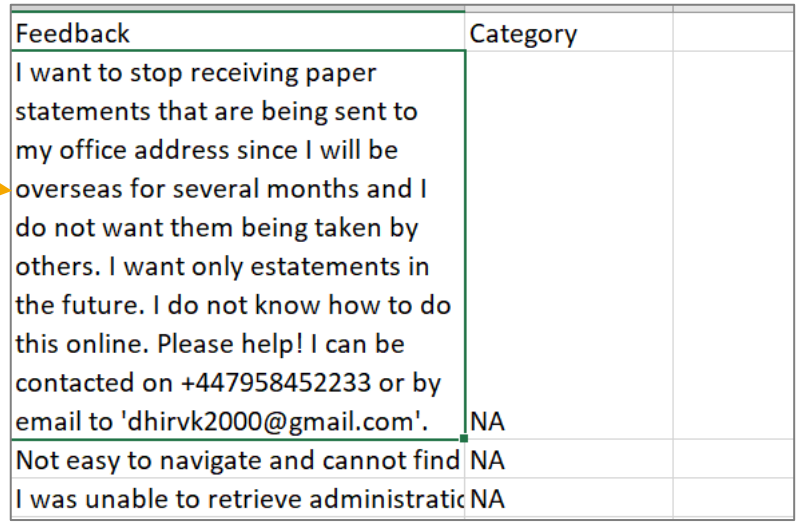



User manual

# 1 Uploading your file

- Place your raw feedbacks into an excel file
- Following the instructions to prepare your file:
    I. Create a column for your feedback and header as “Feedback”
    II. Create a column header: “Category” and fill it with “NA”
    III. Save your file as CSV-UTF8 (Comma Delimited)

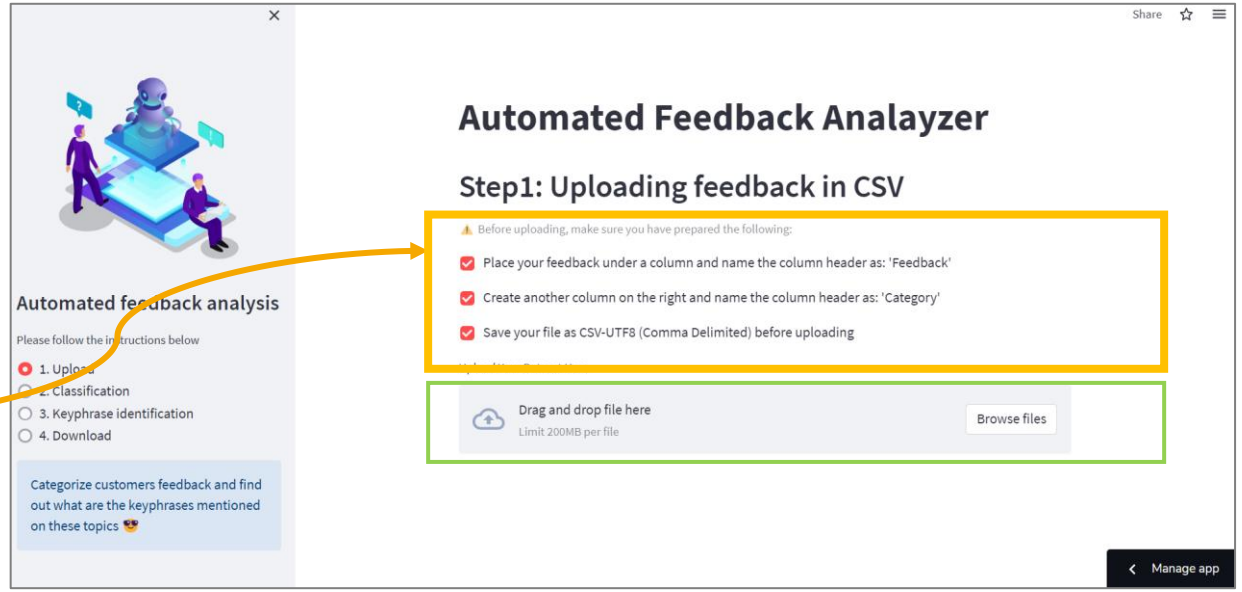


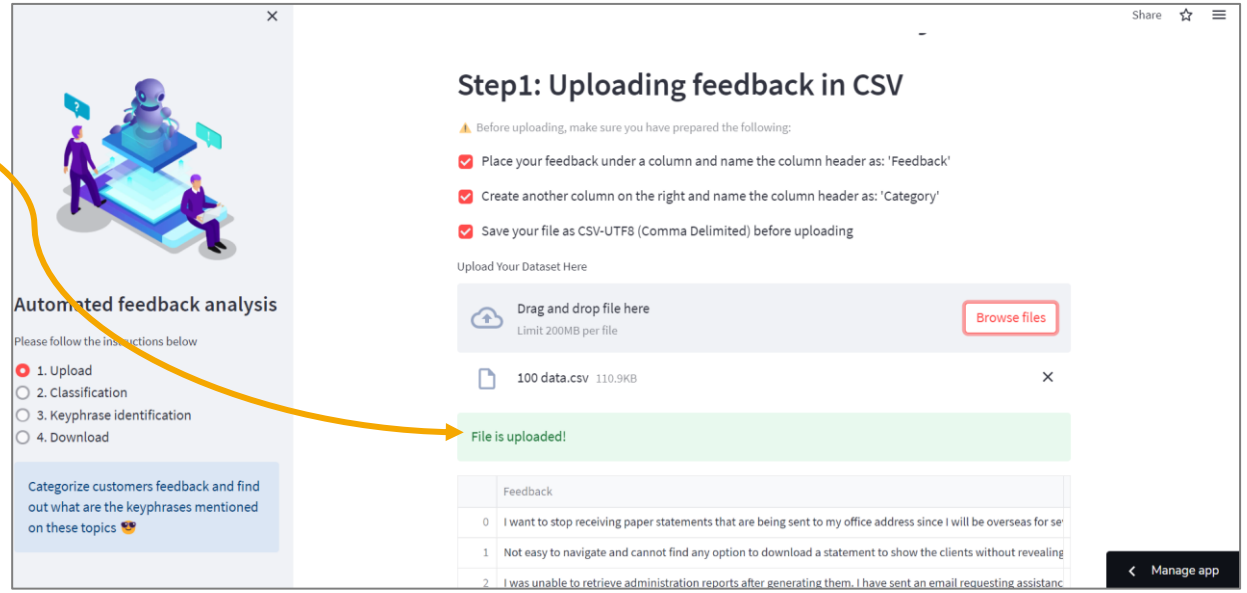



| Feedback | Category |
|---|---|
| I want to stop receiving paper statements that are being sent to my office address since I will be overseas for several months and I do not want them being taken by others. I want only estatements in the future. I do not know how to do this online. Please help! I can be contacted on +447958452233 or by email to 'dhirvk2000@gmail.com'. | NA |
| Not easy to navigate and cannot find | NA |
| I was unable to retrieve administratic | NA |

# 1 Uploading your file

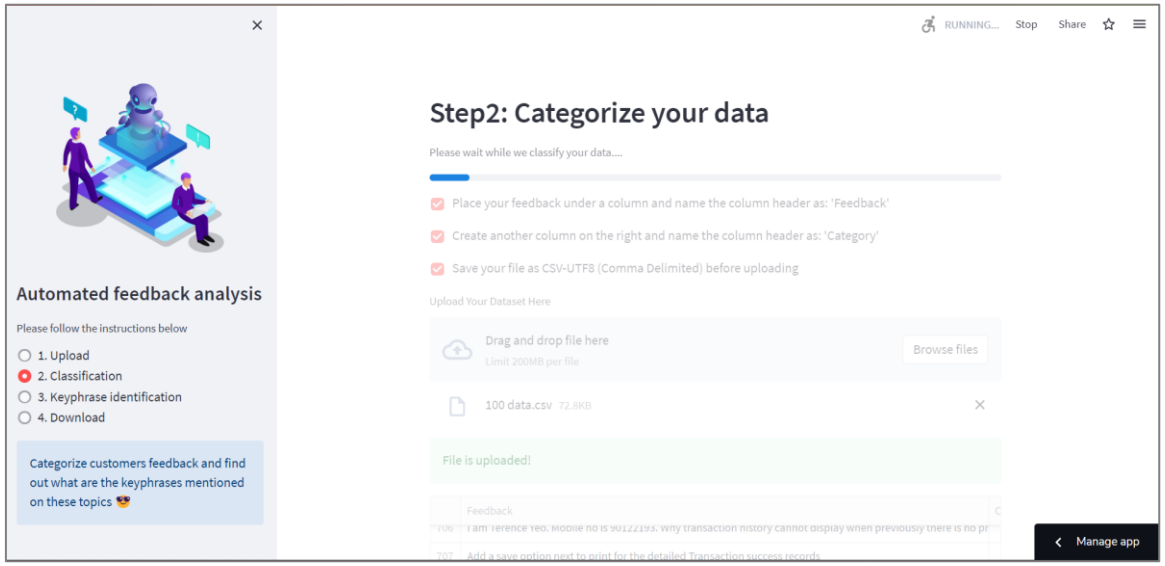
- To enable uploading, check all boxes
- Upload your file into “Drag and drop file here”
- File is uploaded
- Click “2. Classification” to proceed

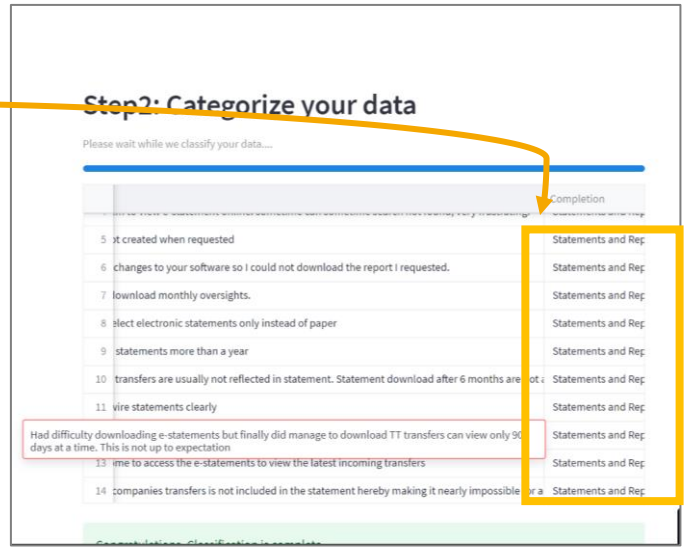


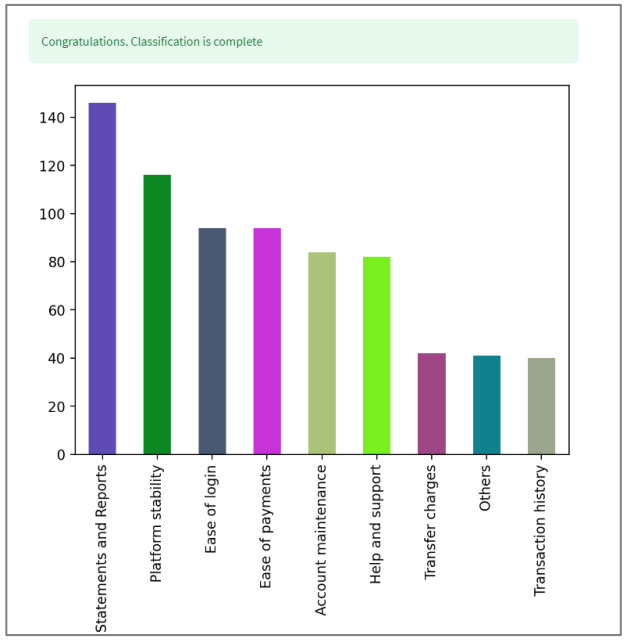



# 2 Categorizing data

- Once “2. Classification” is selected, the categorizing begins as shown on the progress bar
- 500-700 feedback max 30 minutes (can change tab while waiting)
- Once categorization finishes, the categories will be shown with its respective feedback
- A bar graph with counts of each category will be shown
- Click “3. Keyphrase Identification” to proceed

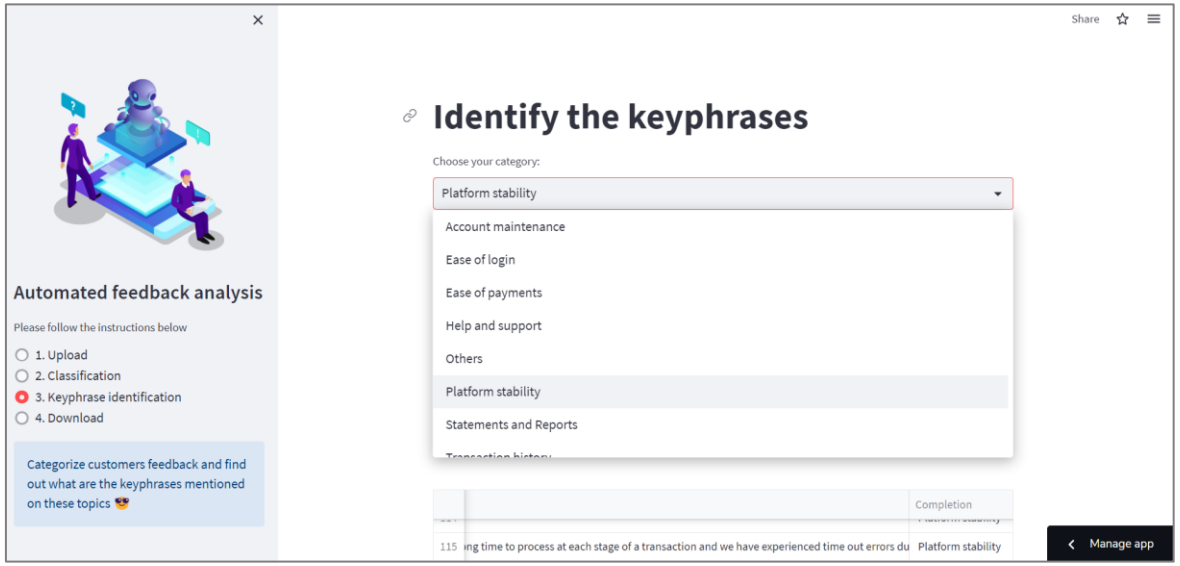


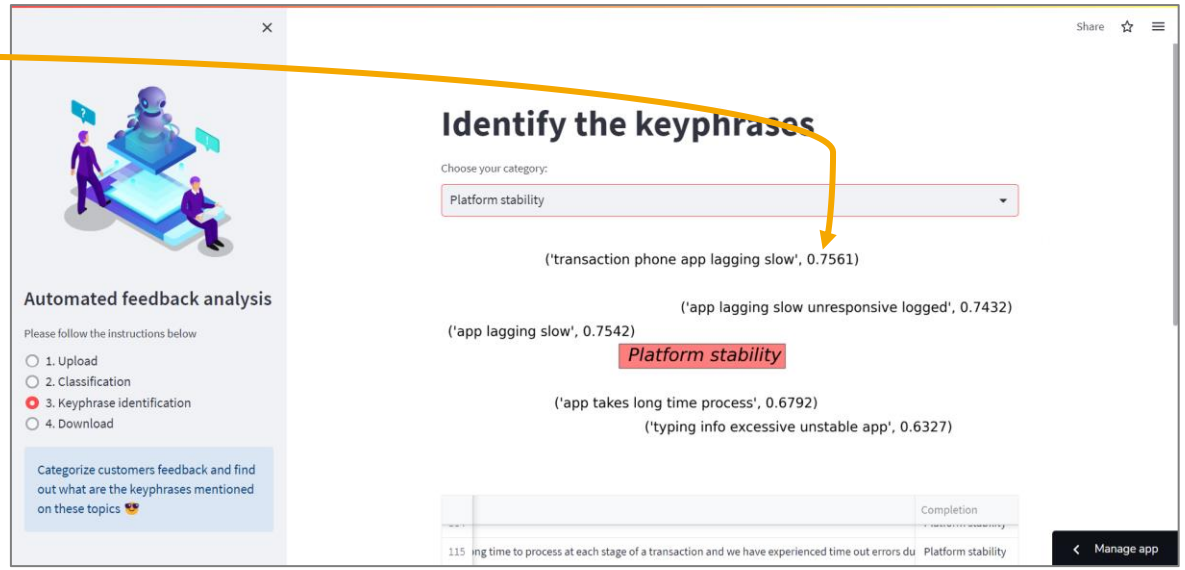



# 3 Keyphrases

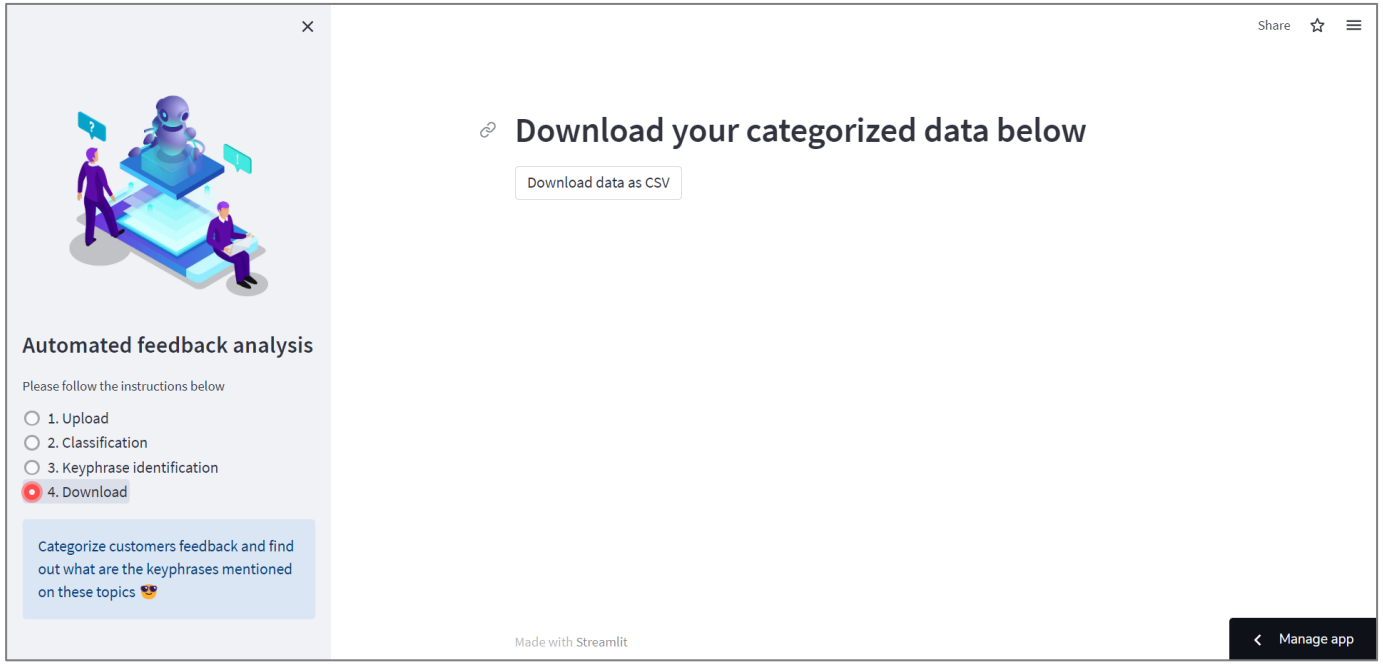
- Once “3. Keyphrase identification” is selected, key phrases are generated
- From the dropdown, select the category (8 categories to choose from)
- The top 5 key phrases will be shown
- Please ignore the index shown on the right of each key phrase
- Click “4. Download” to proceed



# 4 Download file

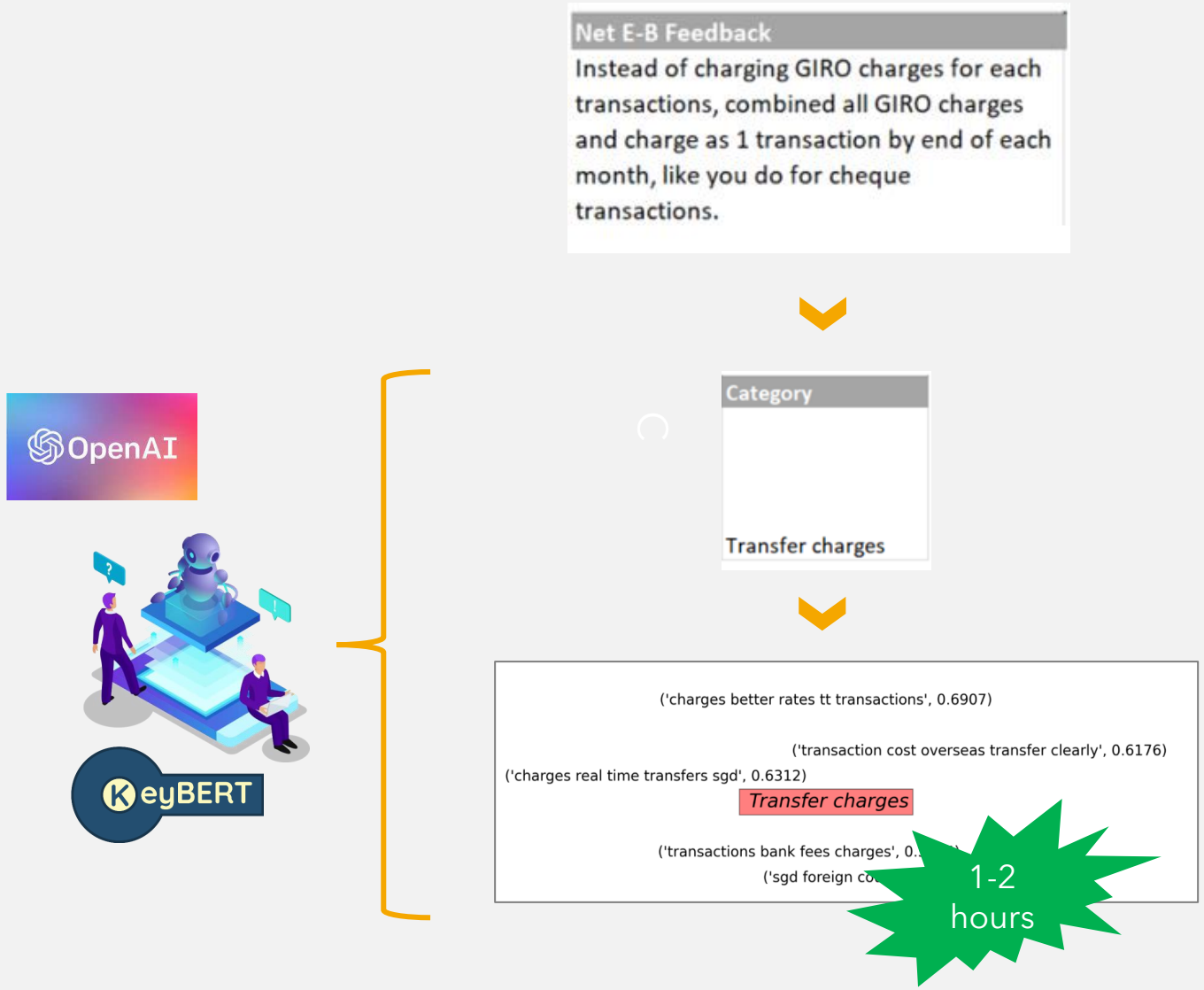
- Once “4. Download” is selected, the categorized file is ready for download
- Click the “Download” button and it will be downloaded in CSV format
- File will be named as “Completeclassifications”



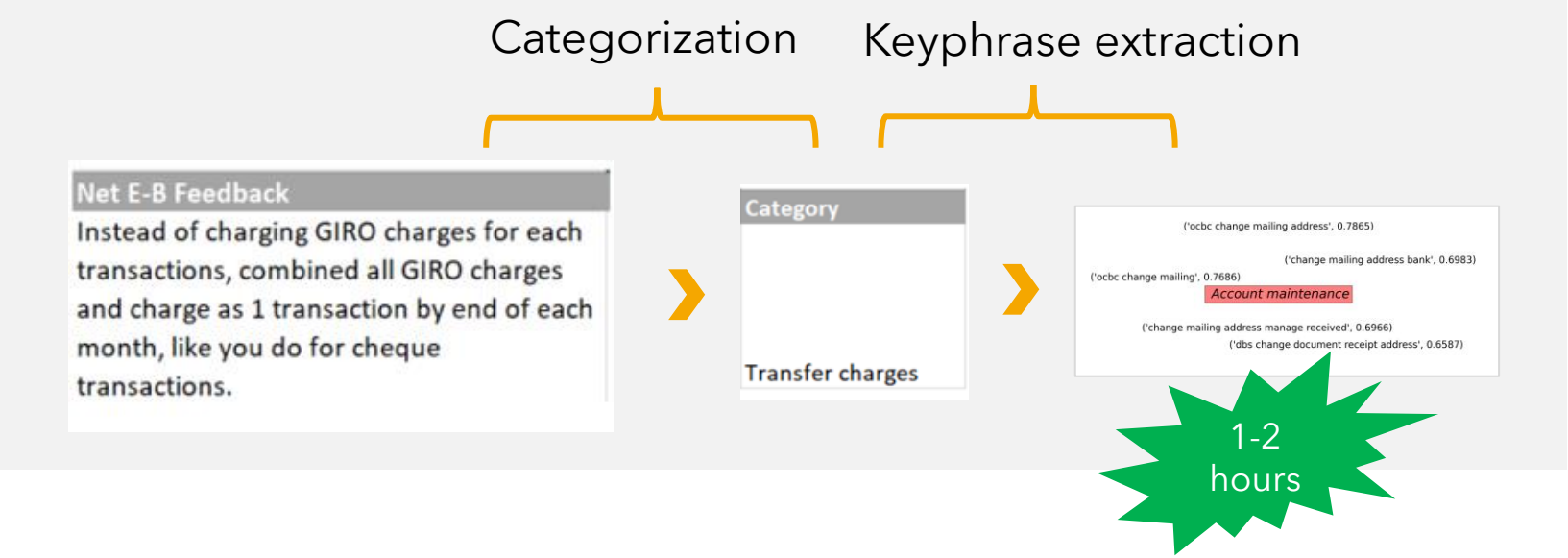

| | Prompt | Completion |
|---|---|---|
| 0 | I want to stop receiving paper | Statements and Reports |
| 1 | Not easy to navigate and cann | Statements and Reports |
| 2 | I was unable to retrieve admi | Statements and Reports |
| 3 | I can not download e- | Statements and Reports |
| 4 | Always has problem to view e | Statements and Reports |
| 5 | Reports always not created | Statements and Reports |
| 6 | There were some changes to | Statements and Reports |
| 7 | Make it easier to download m | Statements and Reports |
| 8 | need options to select electro | Statements and Reports |

# Concepts

*Understanding the workings, for improvements in the future*



# An Overview

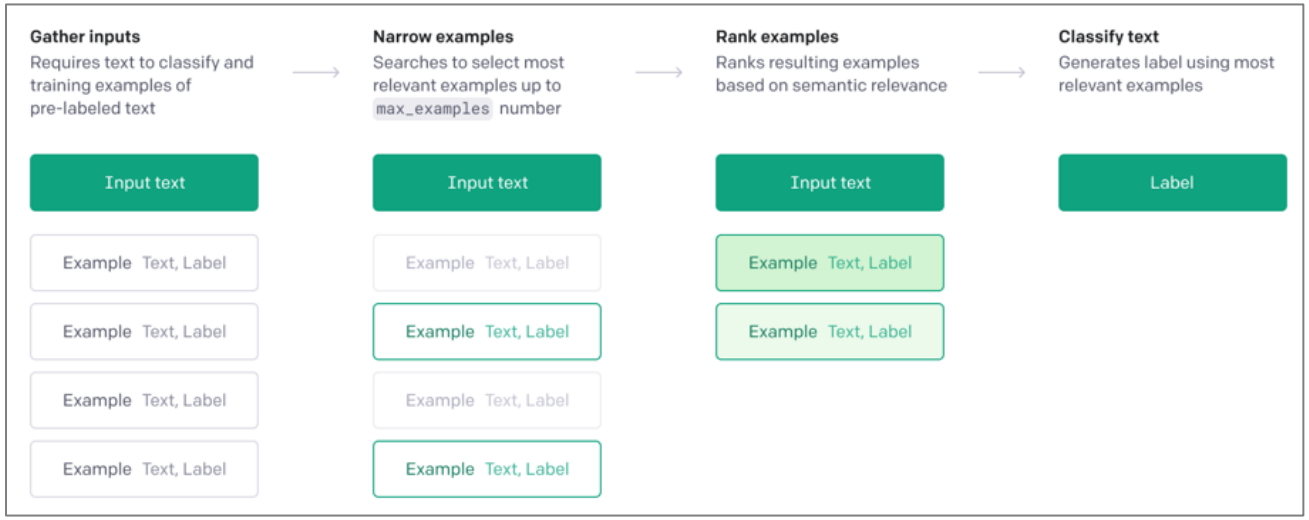



- This web app was run using python code and the full code can be found on Github: https://github.com/JeromeSoh/Feedbackanalysis
- Categorization uses **OpenAI GPT3 model**
- Keyphrase extraction uses **KeyBERT model**
- Website deployment through **Streamlit**

# Categorization

- Utilising OpenAI's text understanding capabilities, we provide examples of text-label pairs to train the model
- This is a process called **Fine-Tuning**. Compared to training our own model using 100k data, Fine-tuning trains a trained-model with minimal examples
- By giving it examples, it learns from this examples so that when similar texts is inputted in the future, it will return the category it was trained on


**ChatGPT** is a conversational chatbot that was launched by **OpenAI** in November 2022. Trained on billions of text data, It has strong dialogue capabilities and is smart enough to answer coding questions



# Categorization

- The current model has been trained to classify feedback based on these 8 categories:

1. Statements and Reports
2. Ease of login
3. Account maintainance
4. Help and support: Customer service related issues
5. Ease of payments: Managing payees, templates, repeated payments
6. Transfer charges
7. Transaction history
8. Others: Trade, letters of credits, invoices, card limits etc.

# Categorization

```
for index, row in df1.iterrows():
    from tenacity import (
        retry,
        stop_after_attempt,
        wait_random_exponential,
    )  # for exponential backoff

    @retry(wait=wait_random_exponential(min=1, max=60), stop=stop_after_attempt(20))
    def completion_with_backoff(**kwargs):
        return openai.Completion.create(**kwargs)

    prompt = row["Prompt"]
    # Send the prompt to the model
    openai.api_key = st.secrets["api_key"]
    response = completion_with_backoff (
        model= "ada:ft-singapore-polytechnic-2023-01-24-03-28-17",
        prompt= prompt + "\n\nIntent:\n\n",
        max_tokens=5,
        temperature=0,
        top_p=1,
        frequency_penalty=0,
        presence_penalty=0,
        stop=[" END"]
    )
    response=response['choices'][0]['text']
    # Append the completion to the completions list
    completions.append(response)
    # Update the progress bar
    progress_bar.progress((index+1)/ total_rows)
```

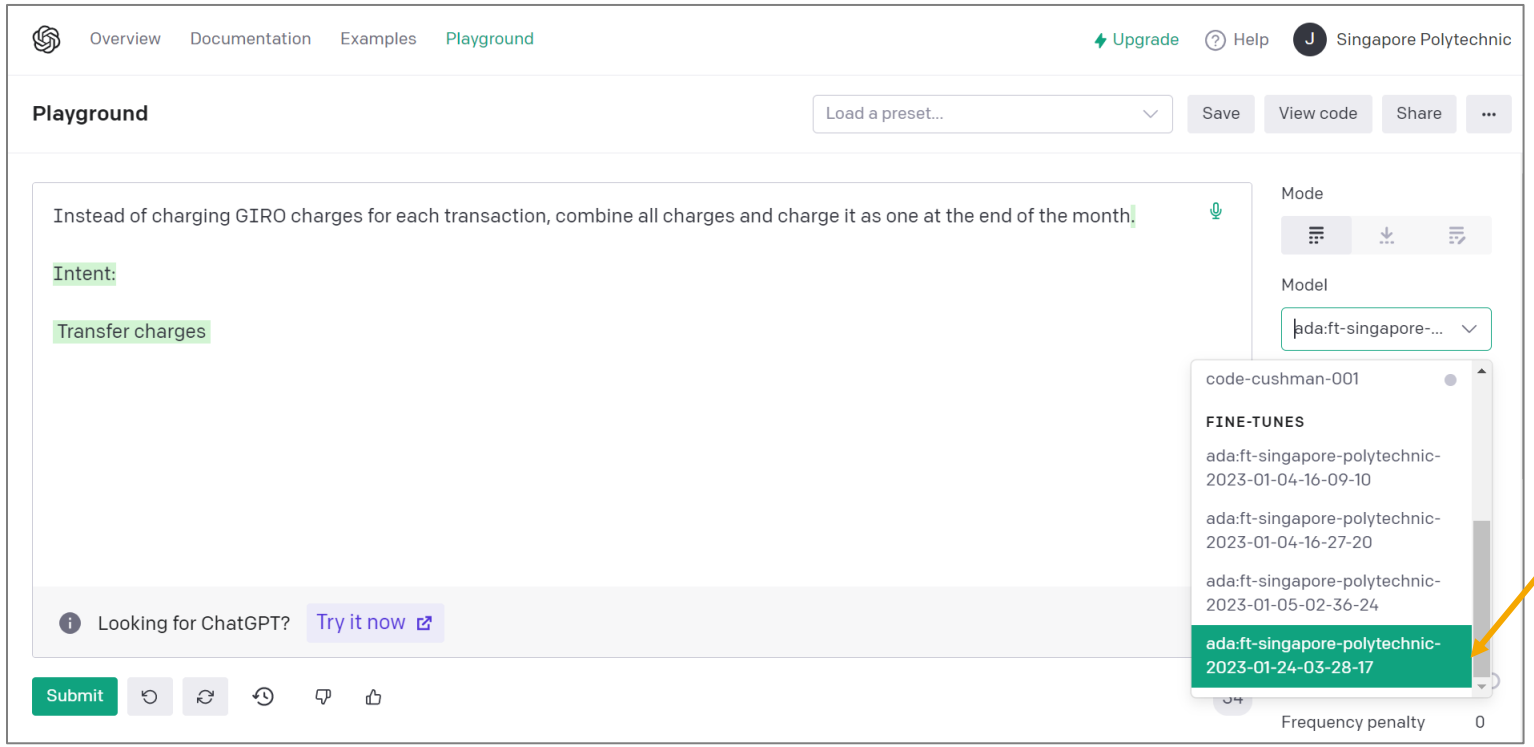
*Let's understand the code:*

- For each row in the file,
- Exponential Backlog: Every time the app hits API rate limit error,it automatically waits and resends
- Send the API request to my fine-tuned model called "ada:ft-Singapore-polytechnic – 2023 -01-24-03-28-17"
- Receive the category and add to respective feedback

This means that for each row of feedback, the app sends the feedback through an API request to your fine-tuned model, and receives a category. If you have 500 rows, it repeats 500 times.

# Categorization

- To fine tune your own model, sign up with OpenAI to create & store your model and generate API keys
- API keys allow you to access your OpenAI models
- After fine-tuning your model, you can test it on a playground environment
- For more information, visit: https://platform.openai.com/docs/guides/completion/prompt-design




This is my playground environment:

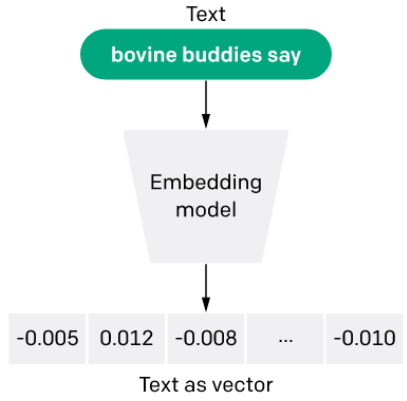
- I can input any feedback and see what response it returns
- Using the model that I fine tuned

# Keyphrase Extraction

- This feature uses the KeyBERT model, these explained are good-to-knows:

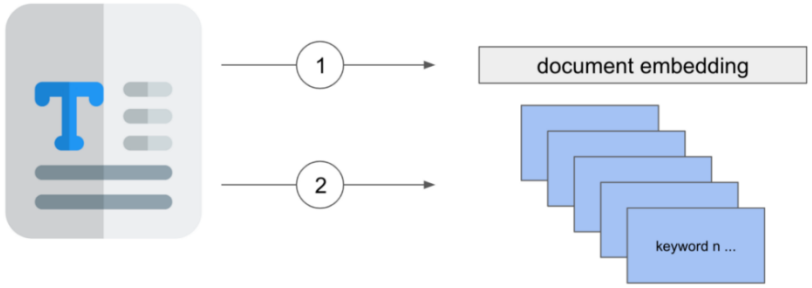

## 1. Document embedding

The feedback is passed into a pre-trained BERT model which turns the text into numbers that is used to represent the feedback as a whole




## 2. Phrase extraction

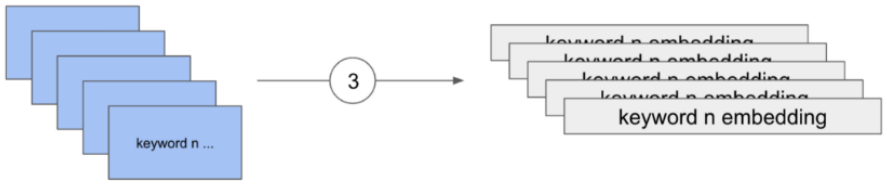
Phrases from the feedback are extracted using Bag of Words technique, which takes the frequency of words



# Keyphrase Extraction

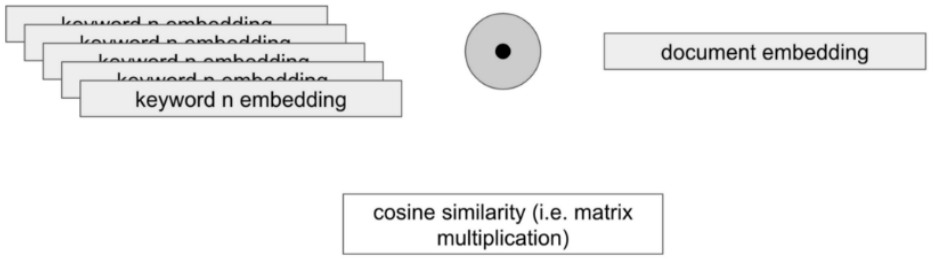
## 3. Phrase embedding

The phrases are vectorized based on the embeddings made by the feedback in part one




## 4. Key-Phrase extraction

Now that the frequency of key phrases have been counted **(2. Phrase extraction)** and it has been assigned a numerical value based on the feedback sentence **(3. Phrase embedding)**, it can be compared with the feedback sentence **(1. Document embedding)** to find out how important the phrase is within the document. Hence, the name **"Key" phrase extraction.**




*To find out more, visit:*
*https://towardsdatascience.com/how-to-extract-relevant-keywords-with-keybert-6e7b3cf889ae*

Concepts

# Keyphrase Extraction

- This were the results of using the KeyBERT model with two different parameters, can you spot the difference?:

```
doc = ["""I have been trying to change mailing address and keep telling me error.
Ibanking could not do the hcange of mailing and registered address of the company.
Could not find a change the mailing address option.
Could not change address online. Reason unknown.
The address change option exited with a system error message.
Please make this link actually work when your customer wants to make an address change: https://www.XXXX.com/business-bankin
keep in one function for change address and change contact.
"Ability to make account changes without filling out the paperwork and waiting days or loner for ""approval"". If the XXXX O
Ability to change Corporate details such as details.
Change of mailing address via email.
```

```
#Max-sum distance:
#To diversify the results, we take the 2 x top_n most similar words/phrases to the document.
#Then, we take all top_n combinations from the 2 x top_n words and extract the combination that are the
kw_model.extract_keywords(doc, keyphrase_ngram_range=(3, 5), stop_words='english',
                          use_maxsum=True, nr_candidates=20, top_n=5)
```

```
[('registered address company change', 0.5738),
 ('error ibanking hcange mailing registered', 0.5778),
 ('trying change mailing', 0.5842),
 ('address change option exited error', 0.5952),
 ('mailing address telling error ibanking', 0.5993)]
```

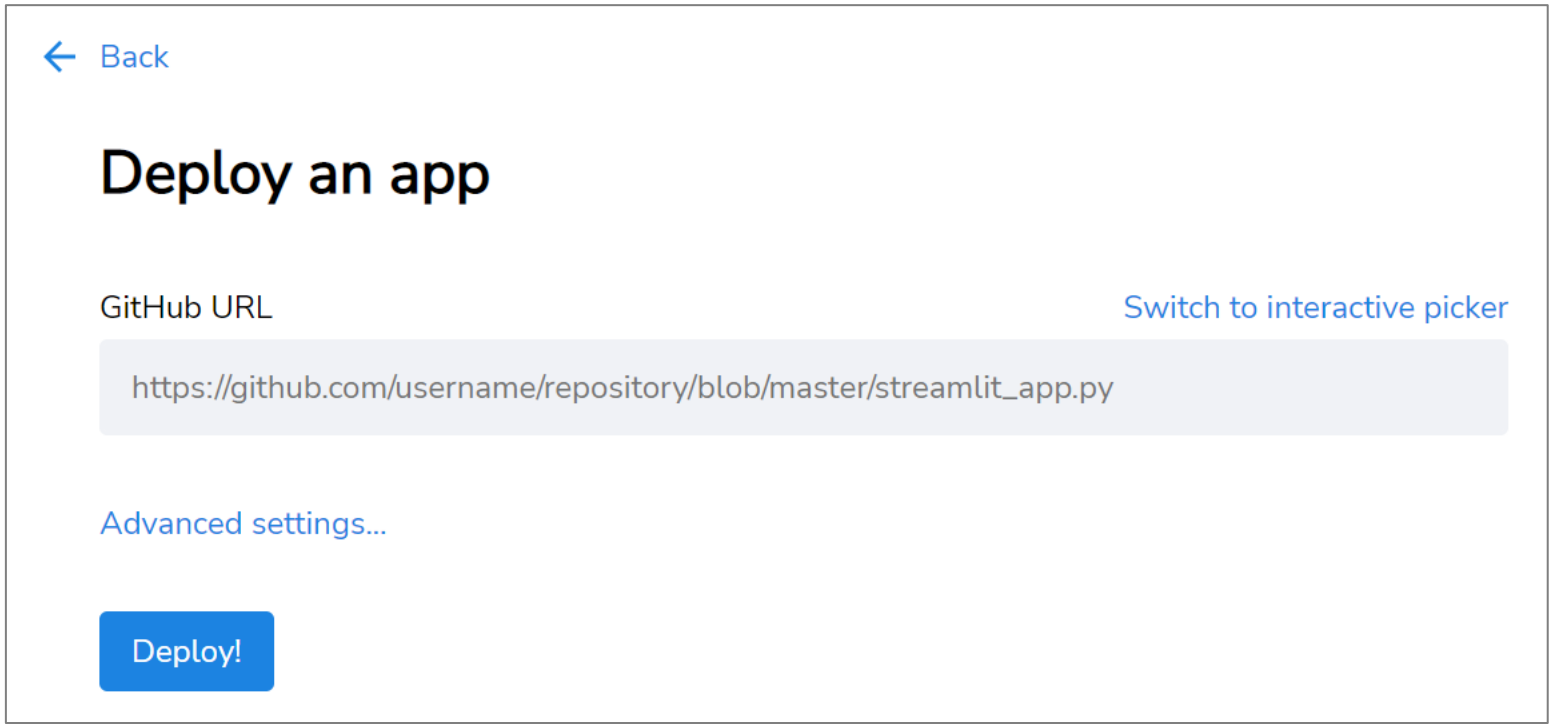
```
#Maximal marginal reference
#Use cosine similarity to have a high diversity
kw_model.extract_keywords(doc, keyphrase_ngram_range=(3, 5), stop_words='english',
                          use_mmr=True, diversity=0.7)
```

```
[('ocbc com business banking help', 0.592),
 ('change mailing address telling error', 0.5768),
 ('mobile paperwork excessive ability', 0.2782),
 ('help support section forms function', 0.0875),
 ('days loner approval', 0.0563)]
```

- As you can see, the one in blue has more diverse key phrases than the one in yellow. This parameter is called “Diversity”

# Deploying the Web app

- Using **Streamlit,** Machine Learning apps can be easily deployed publicly
- By linking the Github to Streamlit, they are able to convert the app within minutes
- There are various built in features, so developers do not have to use traditional HTML or CSS code to create the website
- Its free too!



# Future Suggestions

*The current model is able to classify 500-700 rows within 30 minutes and can be run in the background browser (you can do other things while waiting). However, there are improvements that can be made:*

1) **Retraining of the classifier**

   The main understanding is that the model could only classify feedback that it was taught on. However, the current model was trained using the data in 2020 to 2021. If a new feature was rolled out during 2022 to 2023, it might not be able to classify the feedback accurately. Hence, I suggest that retraining using new data and updating the training categories will need to be carried out every now and then so that the model is up to date.

2) **Sending batch requests**

   Instead of sending each API request individually, I suggest to train the model to take in batch requests by providing examples of batches to the model. This is to save computational power and improve the speed of the analysis.

# Future Suggestions

*The current model is able extract key phrases. But are these key phrases, able to represent the feedback well? What can we do to make the app even better?*

**3) Data cleaning**

Before the feedback gets analysed, we want to remove any meaningless feedback. Example of such feedback where no information can be deduced from it included “sdjkkjdsn not working”, “why not working today…why”, “lame”. This is to not waste time to classify them and affect the keyword extraction. To do this, I suggest to carry out a study on how customers would phrase these type of feedbacks and add a feature within the web app to filter out these types of feedbacks before analysing it.

**4) Improving the keyword extraction**

The extraction of key phrases can be hard to understand and inaccurate at times. I suggest to explore alternatives besides the keyBERT key phrase extraction as there can be better solution out there

# Links to follow:

- To access the public web app, visit: https://thefeedbackbot.streamlit.app/
- To access the public code, visit: https://github.com/JeromeSoh/Feedbackanalysis
- To fine-tune your own model, visit: https://platform.openai.com/docs/guides/completion/prompt-design
- To learn more about KeyBERT, visit: https://towardsdatascience.com/how-to-extract-relevant-keywords-with-keybert-6e7b3cf889ae
- To deploy your own streamlit app, visit: https://docs.streamlit.io/streamlit-community-cloud/get-started/deploy-an-app