Importing the Required Libraries

```
!pip install matplotlib
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (3.7.1)
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.24.3)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (0.11.0)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (8.4.0)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (4.39.3)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (3.0.9)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.4)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (23.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.0.7)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
```

```
!pip install --upgrade pandas
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (2.0.1)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2022.7.1)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-packages (from pandas) (1.24.3)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2023.3)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
```

```
import json
import glob
import nltk
import spacy
import gensim
import gensim.corpora as corpora
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from gensim.utils import simple_preprocess
from gensim.models import CoherenceModel
from nltk.corpus import stopwords
from wordcloud import WordCloud
```

Preparing the Data

```
df = pd.read_excel('/content/drive/MyDrive/Project Data.xls')
print(df.head(5))
```

```
                                      ConversationId  \
0  AAQkAGVjMmUyMDljLTAyMzYtNGU0MC1iOGFmLWI3NWQzM2...
1  AAQkAGVjMmUyMDljLTAyMzYtNGU0MC1iOGFmLWI3NWQzM2...
2  AAQkAGVjMmUyMDljLTAyMzYtNGU0MC1iOGFmLWI3NWQzM2...
3  AAQkAGVjMmUyMDljLTAyMzYtNGU0MC1iOGFmLWI3NWQzM2...
4  AAQkAGVjMmUyMDljLTAyMzYtNGU0MC1iOGFmLWI3NWQzM2...

                                          Subject  \
0          5130 Browns Pt Blvd Unit D - Landscaping
1     Fwd: 5130 Browns Pt Blvd Unit D - Landscaping
2      RE: 5130 Browns Pt Blvd Unit D - Landscaping
3  File Number-3645595-Address-12526 SE 32nd St U...
4  RE: File Number-3645595-Address-12526 SE 32nd ...

                                             Body     Category  \
0  Dear Propvivo, \r\nThank you for the recent le...  Maintenance
1  Sorry , here are the pictures. [cid:0DD370C3-6...  Maintenance
2  Dear Russ & Shanna,\r\n\r\nWe Thank You for yo...  Maintenance
3  \r\n\r\n\r\n\r\n\r\n\r\nFile No.: 4244 3645595\r\n...        Other
4  Hi,\r\n\r\nCan you please confirm if this a Ru...      Payment

   HasAttachment    DateTimeReceived    UnitNumber
0          False 2019-10-09 20:02:06  Unit # 5130D
1          False 2019-10-09 20:10:03  Unit # 5130D
2          False 2019-10-09 22:31:00  Unit # 5130D
3           True 2021-01-29 05:56:13           NaN
4          False 2021-01-29 21:19:00           NaN
```

```
def load_data(link):
    df = pd.read_excel(link)
    return df



nltk.download('stopwords')
stopwords = stopwords.words("english")
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
print (stopwords)
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself',
```

```
data = load_data('/content/drive/MyDrive/Project Data.xls')["Body"]

#cleaning the data
# Remove empty strings
data = data.replace('', pd.NA)

# Drop rows with null values
data = data.dropna()

print (data[0][0:90])
```

```
Dear Propvivo,
Thank you for the recent letter regarding the dog poop remains in the way
```

```
def lemmatization(texts, allowed_postags=["NOUN", "ADJ", "VERB", "ADV"]):
    nlp = spacy.load("en_core_web_sm", disable=["parser", "ner"])
    texts_out = []
    for text in texts:
        # if isinstance(text, str) and not pd.isna(text):  # Check for NaN and empty strings
          doc = nlp(text)
          new_text = []
          for token in doc:
              if token.pos_ in allowed_postags:
                  new_text.append(token.lemma_)
          final = " ".join(new_text)
          texts_out.append(final)
    return texts_out

lemmatized_body = lemmatization(data)
print(lemmatized_body[0][0:90])
```

```
thank recent letter regard dog poop remain way landscaper make sure future remain pick imm
```

```
def gen_words(texts):
    final = []
    for text in texts:
        new = gensim.utils.simple_preprocess(text, deacc=True)
        final.append(new)
    return (final)

data_words = gen_words(lemmatized_body)

print (data_words[0][0:20])
```

```
['thank', 'recent', 'letter', 'regard', 'dog', 'poop', 'remain', 'way', 'landscaper', 'make', 'sure', 'future', 'remain', 'pick', 'immed
```

```
id2word = corpora.Dictionary(data_words)

corpus = []
for text in data_words:
    new = id2word.doc2bow(text)
    corpus.append(new)

print (corpus[0][0:20])
```

```
word = id2word[[0][:1][0]]
print (word)
```

```
[(0, 1), (1, 1), (2, 1), (3, 2), (4, 1), (5, 1), (6, 1), (7, 2), (8, 1), (9, 4), (10, 1), (11, 1), (12, 1), (13, 2), (14, 1), (15, 1), (
again
```

```
lda_model = gensim.models.ldamodel.LdaModel(corpus=corpus,
                                            id2word=id2word,
                                            num_topics=30,
                                            random_state=100,
                                            update_every=1,
                                            chunksize=100,
                                            passes=10,
                                            alpha="auto")
```

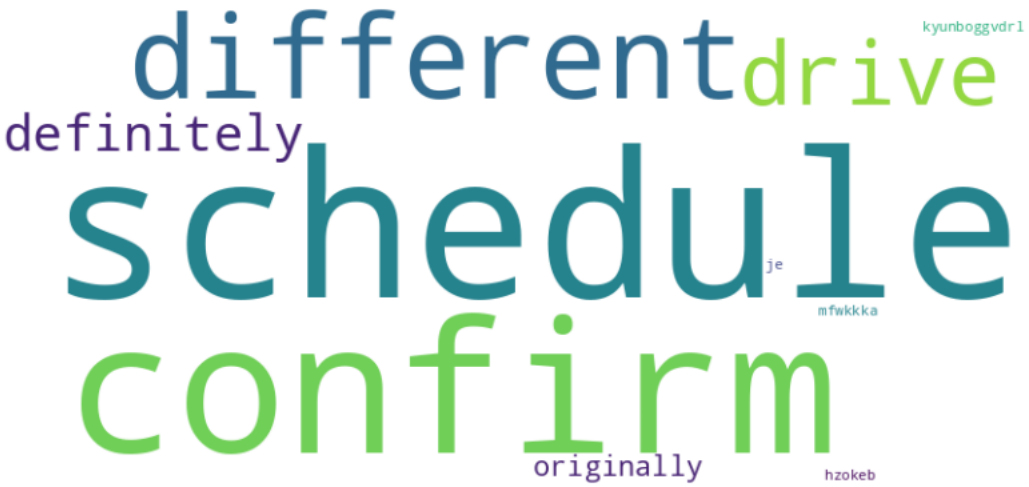
Vizualizing the Data

```
# Print the top 10 keywords in each topic
for topic in lda_model.show_topics(num_topics=30, num_words=10):
    print(topic)
```

```
(0, '0.267*"schedule" + 0.218*"confirm" + 0.053*"different" + 0.028*"drive" + 0.008*"definitely" + 0.001*"originally" + 0.000*"je" + 0.0
(1, '0.118*"meeting" + 0.051*"proxy" + 0.037*"vote" + 0.033*"attend" + 0.025*"send" + 0.022*"enclose" + 0.021*"pm" + 0.021*"person" + 0.
(2, '0.089*"intend" + 0.080*"message" + 0.068*"information" + 0.064*"notify" + 0.050*"use" + 0.047*"immediately" + 0.046*"contain" + 0.0
(3, '0.224*"cid" + 0.170*"image" + 0.150*"detail" + 0.118*"png" + 0.096*"portal" + 0.026*"ad" + 0.020*"ff" + 0.014*"violation" + 0.014*"
(4, '0.323*"board" + 0.093*"visit" + 0.090*"available" + 0.078*"much" + 0.045*"answer" + 0.037*"agreement" + 0.033*"member" + 0.032*"agr
(5, '0.074*"notice" + 0.051*"today" + 0.035*"click" + 0.034*"monthly" + 0.029*"try" + 0.026*"here" + 0.026*"then" + 0.026*"mail" + 0.022
(6, '0.127*"approval" + 0.127*"then" + 0.097*"very" + 0.078*"move" + 0.058*"leave" + 0.040*"remind" + 0.038*"parking" + 0.034*"location"
(7, '0.297*"complete" + 0.081*"here" + 0.075*"mention" + 0.075*"directly" + 0.057*"dd" + 0.007*"removal" + 0.007*"return" + 0.000*"uploa
(8, '0.193*"com" + 0.133*"https" + 0.094*"pay" + 0.092*"www" + 0.075*"http" + 0.045*"png" + 0.034*"cc" + 0.028*"jpg" + 0.022*"images" +
(9, '0.060*"nd" + 0.000*"hqgfaceycmuha" + 0.000*"fuglazyjzkvifn" + 0.000*"nkcu" + 0.000*"mfwkkka" + 0.000*"laird" + 0.000*"kyunboggvdrl"
(10, '0.124*"need" + 0.066*"make" + 0.055*"time" + 0.052*"take" + 0.048*"go" + 0.046*"include" + 0.045*"day" + 0.037*"change" + 0.030*"p
(11, '0.067*"email" + 0.062*"document" + 0.030*"sign" + 0.026*"docusign" + 0.023*"https" + 0.019*"link" + 0.018*"sender" + 0.016*"net" +
(12, '0.061*"de" + 0.042*"jjtrmfo" + 0.042*"profile" + 0.034*"image" + 0.032*"mc" + 0.031*"di" + 0.030*"iogfmlwi" + 0.030*"fxpeqzuyqhgye
(13, '0.151*"week" + 0.143*"homeowner" + 0.127*"just" + 0.114*"next" + 0.083*"there" + 0.074*"project" + 0.043*"box" + 0.013*"at" + 0.00
(14, '0.120*"file" + 0.114*"unit" + 0.114*"fee" + 0.065*"close" + 0.047*"ce" + 0.043*"open" + 0.038*"place" + 0.034*"fd" + 0.032*"be" +
(15, '0.137*"year" + 0.087*"last" + 0.070*"response" + 0.069*"tree" + 0.055*"come" + 0.033*"roof" + 0.031*"consider" + 0.030*"early" + 0
(16, '0.106*"send" + 0.085*"email" + 0.053*"th" + 0.037*"work" + 0.032*"letter" + 0.031*"provide" + 0.031*"owner" + 0.031*"receive" + 0.
(17, '0.108*"job" + 0.062*"sheet" + 0.051*"order" + 0.049*"track" + 0.044*"email" + 0.039*"receive" + 0.032*"follow" + 0.027*"contact" +
(18, '0.175*"message" + 0.059*"subject" + 0.057*"recipient" + 0.037*"sender" + 0.034*"pass" + 0.034*"com" + 0.030*"securely" + 0.028*"or
(19, '0.223*"reply" + 0.090*"mailto" + 0.078*"db" + 0.034*"dc" + 0.000*"managemowed" + 0.000*"kyunboggvdrl" + 0.000*"laird" + 0.000*"je"
(20, '0.162*"thank" + 0.097*"know" + 0.093*"let" + 0.068*"question" + 0.067*"call" + 0.062*"reach" + 0.059*"kindly" + 0.032*"cid" + 0.03
(21, '0.076*"account" + 0.049*"office" + 0.038*"note" + 0.035*"list" + 0.032*"communication" + 0.031*"violation" + 0.030*"as" + 0.027*"f
(22, '0.134*"service" + 0.118*"contact" + 0.104*"ask" + 0.083*"more" + 0.051*"forward" + 0.048*"free" + 0.048*"correct" + 0.045*"feel" +
(23, '0.283*"payment" + 0.244*"check" + 0.119*"invoice" + 0.048*"get" + 0.033*"appreciate" + 0.025*"aka" + 0.012*"speak" + 0.011*"https"
(24, '0.121*"time" + 0.072*"pm" + 0.071*"hour" + 0.067*"cost" + 0.053*"turnaround" + 0.048*"am" + 0.045*"day" + 0.043*"business" + 0.039
(25, '0.362*"address" + 0.088*"again" + 0.052*"building" + 0.050*"mailing" + 0.030*"line" + 0.029*"confirm" + 0.029*"actual" + 0.026*"cl
(26, '0.154*"policy" + 0.059*"term" + 0.058*"material" + 0.046*"condition" + 0.019*"ongoing" + 0.015*"mailbox" + 0.003*"cg" + 0.000*"omc
(27, '0.164*"pende" + 0.083*"mail" + 0.063*"review" + 0.054*"recipient" + 0.053*"below" + 0.049*"error" + 0.044*"no" + 0.039*"long" + 0.
(28, '0.047*"click" + 0.043*"upn" + 0.036*"net" + 0.032*"proposal" + 0.031*"ct" + 0.031*"ls" + 0.030*"gdw" + 0.030*"ndf" + 0.030*"sendgr
(29, '0.110*"good" + 0.054*"financial" + 0.049*"sure" + 0.046*"meet" + 0.042*"create" + 0.038*"way" + 0.035*"few" + 0.030*"repair" + 0.0
```

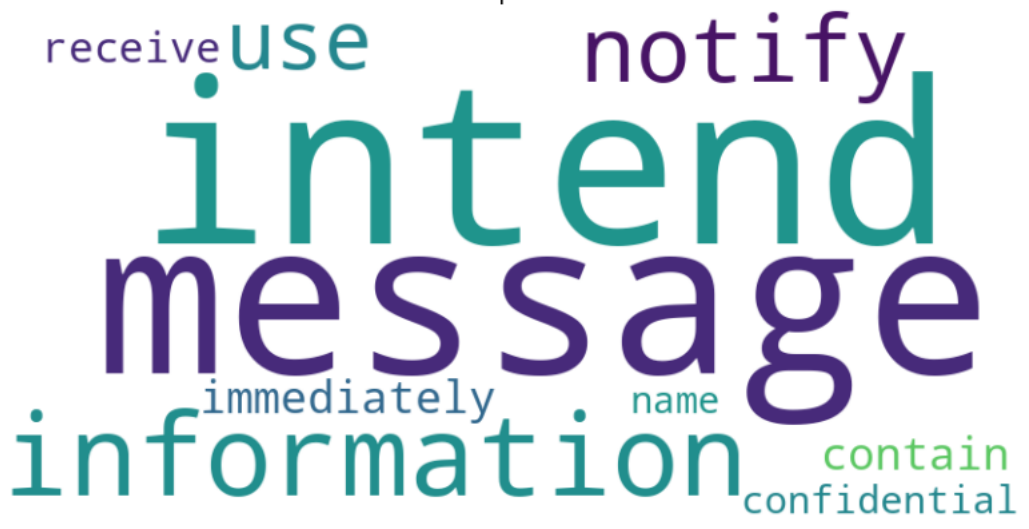
```
# Create a directory to store the images
if not os.path.exists("wordclouds"):
    os.makedirs("wordclouds")

# Create a word cloud for each topic
for i, topic in lda_model.show_topics(num_topics=30, num_words=10, formatted=False):
    topic_words = dict(topic)
    wc = WordCloud(width=800, height=400, background_color="white")
    wc.generate_from_frequencies(topic_words)
    plt.figure(figsize=(10, 5))
    plt.imshow(wc, interpolation="bilinear")
    plt.axis("off")
    plt.title("Topic " + str(i))
    plt.savefig("wordclouds/topic" + str(i) + ".png", bbox_inches="tight")
    plt.show()
```

Topic 0



Topic 1

Topic 2



Topic 3

```
all_keywords = []

for i in range(30):
    keywords = lda_model.get_topic_terms(i, topn=10)
    all_keywords.extend([id2word[id] for id, _ in keywords])

len(all_keywords)
```

```
300
```

```
# Get word-topic probability distribution for each topic
word_probs = []
for topic in lda_model.show_topics(num_topics=30, num_words=100, formatted=False):
    word_probs.append(dict(topic[1]))

# Sum the probabilities for each word across all topics
word_importance = {}
for topic in word_probs:
    for word, prob in topic.items():
        if word in word_importance:
            word_importance[word] += prob
        else:
            word_importance[word] = prob

# Sort the words by their importance in the model
sorted_words = sorted(word_importance.items(), key=lambda x: x[1], reverse=True)

# Create a bar chart of the top 50 words
top_words = sorted_words[:30]
words = [w[0] for w in top_words]
importance = [w[1] for w in top_words]
plt.bar(words, importance)
plt.xticks(rotation=90)
plt.xlabel("Word")
plt.ylabel("Importance")
plt.title("Word Importance in LDA Model")
plt.show()
```