import libraries

In [21]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import nltk
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.preprocessing import LabelBinarizer
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from wordcloud import WordCloud,STOPWORDS
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize,sent_tokenize
from bs4 import BeautifulSoup
import re,string,unicodedata
from nltk.tokenize.toktok import ToktokTokenizer
from nltk.stem import LancasterStemmer,WordNetLemmatizer
from sklearn.linear_model import LogisticRegression,SGDClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC
from sklearn.metrics import classification_report,confusion_matrix,accuracy_score
from sklearn.model_selection import train_test_split
from string import punctuation
from nltk import pos_tag
from nltk.corpus import wordnet
import keras
from keras.layers import Dense
from keras.models import Sequential
```

preprocessing data

In [22]:

```
import pandas as pd

df = pd.read_csv('fake_job_postings.csv', low_memory=False)
display(df.head())
display(df.info())
```

	job_id	title	location	department	salary_range	company_profile	description	requirem
0	1	Marketing Intern	US, NY, New York	Marketing	NaN	We're Food52, and we've created a groundbreaki...	Food52, a fast-growing, James Beard Award-winn...	Experience with cor management syste
1	2	Customer Service - Cloud Video Production	NZ, , Auckland	Success	NaN	90 Seconds, the worlds Cloud Video Production ...	Organised - Focused - Vibrant - Awesome!Do you...	What we expect you:Your responsib
2	3	Commissioning Machinery Assistant (CMA)	US, IA, Wever	NaN	NaN	Valor Services provides Workforce Solutions th...	Our client, located in Houston, is actively se...	Implement commissioning commissioni
3	4	Account Executive - Washington DC	US, DC, Washington	Sales	NaN	Our passion for improving quality of life thro	THE COMPANY: ESRI – Environmental Systems Rese...	EDUCATION: Bache or Master's in b

	job_id	title	location	department	salary_range	company_profile	description	requirem
4	5	Bill Review Manager	US, FL, Fort Worth	NaN	NaN	SpotSource Solutions LLC is a Global Human Cap...	JOB TITLE: Itemization Review ManagerLOCATION:...	QUALIFICATIONS license in the Sta Te

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17880 entries, 0 to 17879
Data columns (total 18 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   job_id               17880 non-null  int64 
 1   title                17880 non-null  object
 2   location             17534 non-null  object
 3   department           6333 non-null   object
 4   salary_range         2868 non-null   object
 5   company_profile      14572 non-null  object
 6   description          17879 non-null  object
 7   requirements         15184 non-null  object
 8   benefits             10668 non-null  object
 9   telecommuting        17880 non-null  int64 
 10  has_company_logo     17880 non-null  int64 
 11  has_questions        17880 non-null  int64 
 12  employment_type      14409 non-null  object
 13  required_experience  10830 non-null  object
 14  required_education   9775 non-null   object
 15  industry             12977 non-null  object
 16  function             11425 non-null  object
 17  fraudulent           17880 non-null  int64 
dtypes: int64(5), object(13)
memory usage: 2.5+ MB

None
```

In [23]:

```
df.describe()
```

Out[23]:

	job_id	telecommuting	has_company_logo	has_questions	fraudulent
count	17880.000000	17880.000000	17880.000000	17880.000000	17880.000000
mean	8940.500000	0.042897	0.795302	0.491723	0.048434
std	5161.655742	0.202631	0.403492	0.499945	0.214688
min	1.000000	0.000000	0.000000	0.000000	0.000000
25%	4470.750000	0.000000	1.000000	0.000000	0.000000
50%	8940.500000	0.000000	1.000000	0.000000	0.000000
75%	13410.250000	0.000000	1.000000	1.000000	0.000000
max	17880.000000	1.000000	1.000000	1.000000	1.000000

In [24]:

```
df.isna().sum()
```

Out[24]:

	0
job_id	0
title	0
location	346
department	11547

	0
salary_range	15012
company_profile	3308
description	1
requirements	2696
benefits	7212
telecommuting	0
has_company_logo	0
has_questions	0
employment_type	3471
required_experience	7050
required_education	8105
industry	4903
function	6455
fraudulent	0

dtype: int64

In [25]:

```
df.drop(['job_id', 'salary_range'], axis=1, inplace=True)
```

In [26]:

```
df.head(5)
```

Out[26]:

	title	location	department	company_profile	description	requirements	benefits	tele
0	Marketing Intern	US, NY, New York	Marketing	We're Food52, and we've created a groundbreaki...	Food52, a fast-growing, James Beard Award-winn...	Experience with content management systems a m...	NaN	
1	Customer Service - Cloud Video Production	NZ, , Auckland	Success	90 Seconds, the worlds Cloud Video Production ...	Organised - Focused - Vibrant - Awesome!Do you...	What we expect from you:Your key responsibilit...	What you will get from usThrough being part of...	
2	Commissioning Machinery Assistant (CMA)	US, IA, Wever	NaN	Valor Services provides Workforce Solutions th...	Our client, located in Houston, is actively se...	Implement pre-commissioning and commissioning ...	NaN	
3	Account Executive - Washington DC	US, DC, Washington	Sales	Our passion for improving quality of life thro...	THE COMPANY: ESRI – Environmental Systems Rese...	EDUCATION: Bachelor's or Master's in GIS, busi...	Our culture is anything but corporate—we have ...	
4	Bill Review Manager	US, FL, Fort Worth	NaN	SpotSource Solutions LLC is a Global Human Cap...	JOB TITLE: Itemization Review ManagerLOCATION:...	QUALIFICATIONS:RN license in the State of Texa...	Full Benefits Offered	

In [27]:

```
df.fillna('', inplace=True)
```

In [28]:

```
df.isna().sum()
```

Out[28]:

	0
title	0
location	0
department	0
company_profile	0
description	0
requirements	0
benefits	0
telecommuting	0
has_company_logo	0
has_questions	0
employment_type	0
required_experience	0
required_education	0
industry	0
function	0
fraudulent	0

dtype: int64

In [29]:

```
import nltk
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
```

Out[29]:

```
True
```

In [30]:

```
df['text'] = df[['title', 'company_profile', 'description', 'requirements', 'benefits']]
.agg(' '.join, axis=1)

def clean_text(text):
    soup = BeautifulSoup(text, "html.parser")
    text = soup.get_text()
    text = re.sub('\[[^]]*\]', '', text)
    text = re.sub('[^a-zA-Z\s]', '', text).lower()

    # Remove stopwords
    stop = set(stopwords.words('english'))
    text = ' '.join([word for word in text.split() if word not in stop])
```

```
    # Lemmatize
    lemmatizer = WordNetLemmatizer()
    text = ' '.join([lemmatizer.lemmatize(word) for word in text.split()])

    return text

df['text'] = df['text'].apply(clean_text)
display(df[['text', 'fraudulent']].head())
```

	text	fraudulent
0	marketing intern food weve created groundbreak...	0
1	customer service cloud video production second...	0
2	commissioning machinery assistant cma valor se...	0
3	account executive washington dc passion improv...	0
4	bill review manager spotsource solution llc gl...	0

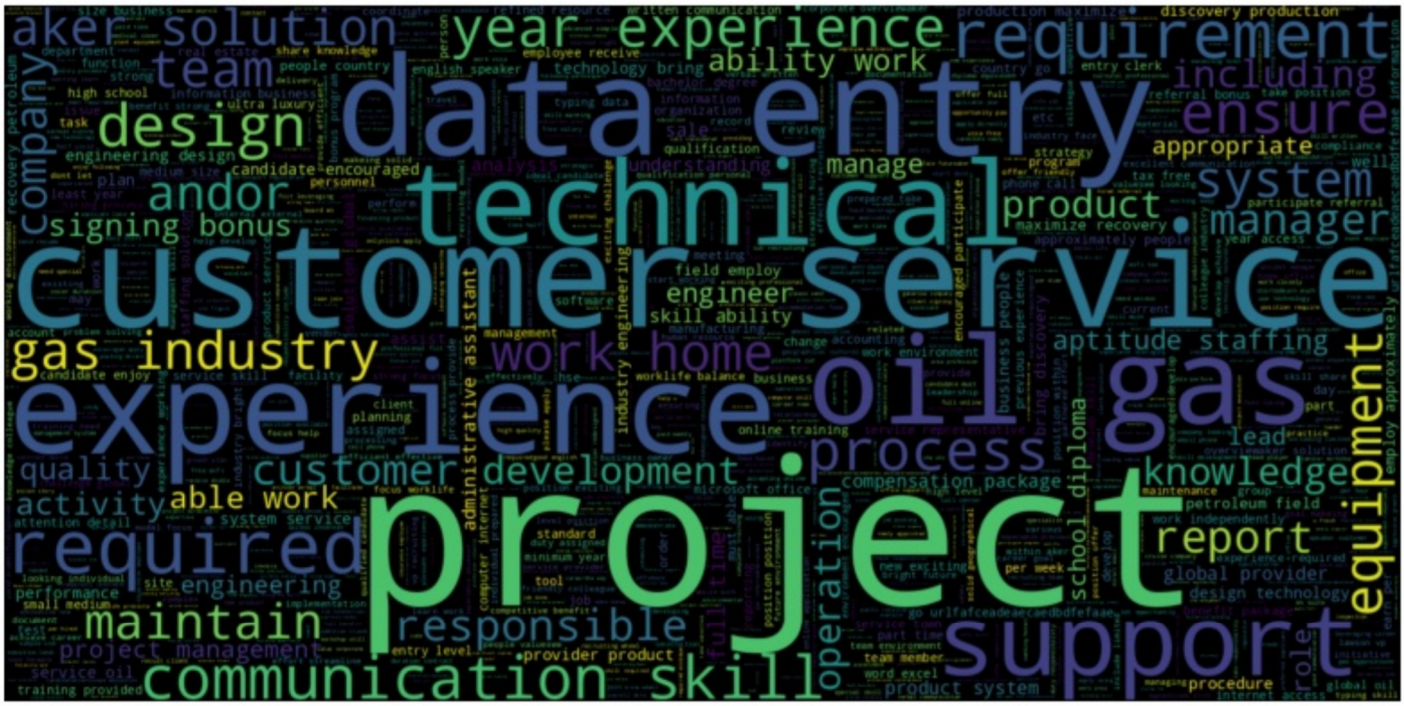
visualize data

In [31]:

```
plt.figure(figsize=(10, 6))
wc = WordCloud(max_words=3000, width=1600, height=800).generate(" ".join(df[df.fraudulen
t == 1].text))
plt.imshow(wc, interpolation='bilinear')
plt.axis("off")
```

Out[31]:

```
(np.float64(-0.5), np.float64(1599.5), np.float64(799.5), np.float64(-0.5))
```



In [32]:

```
plt.figure(figsize=(10, 6))
wc = WordCloud(max_words=3000, width=1600, height=800).generate(" ".join(df[df.fraudulen
t == 0].text))
plt.imshow(wc, interpolation='bilinear')
plt.axis("off")
```

Out[32]:

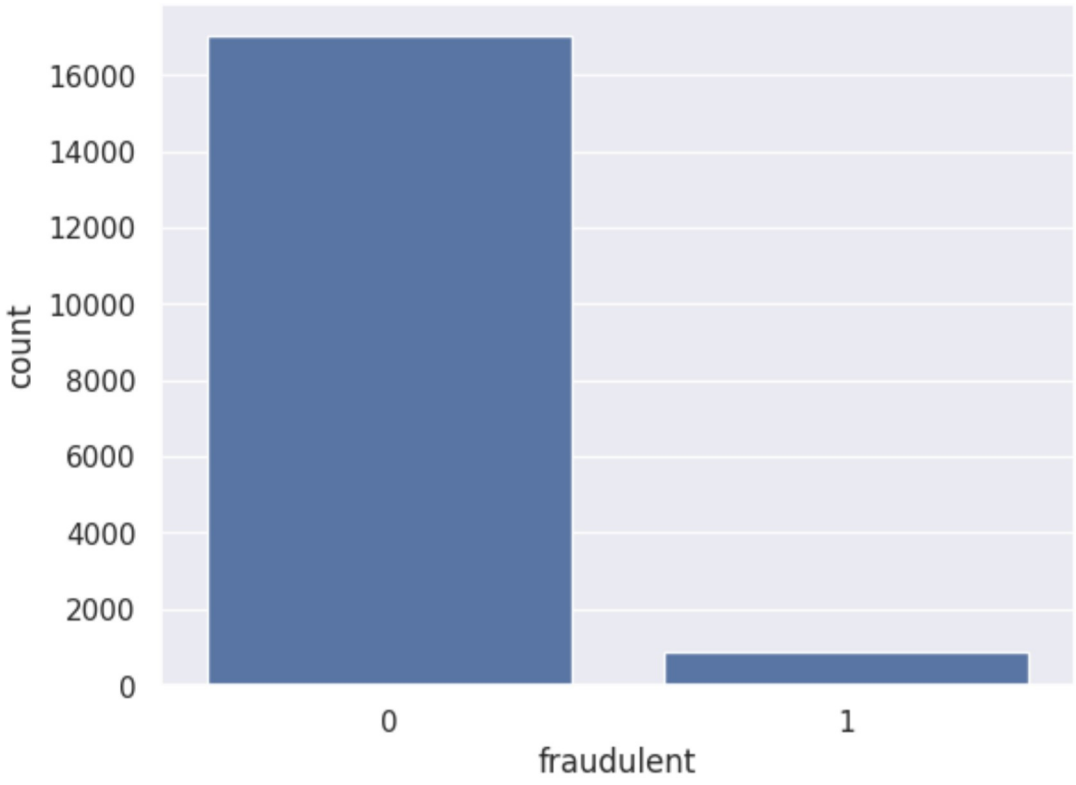
```
(np.float64(-0.5), np.float64(1599.5), np.float64(799.5), np.float64(-0.5))
```

In [33]:

```
sns.set(style="darkgrid")
sns.countplot(x='fraudulent', data=df)
```

Out[33]:

```
<Axes: xlabel='fraudulent', ylabel='count'>
```



train/test split

In [34]:

```
from sklearn.model_selection import train_test_split

X = df['text']
y = df['fraudulent']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42
)

print("Training set shape:", X_train.shape)
print("Testing set shape:", X_test.shape)
```

```
Training set shape: (14304,)
Testing set shape: (3576,)
```

In [35]:

```
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf_vectorizer = TfidfVectorizer(max_features=5000, min_df=5)

X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)

print("Shape of X_train_tfidf:", X_train_tfidf.shape)
print("Shape of X_test_tfidf:", X_test_tfidf.shape)
```

```
Shape of X_train_tfidf: (14304, 5000)
Shape of X_test_tfidf: (3576, 5000)
```

build model

In [37]:

```
from keras.models import Sequential
from keras.layers import Dense

model = Sequential()
model.add(Dense(128, activation='relu', input_shape=(X_train_tfidf.shape[1],)))
model.add(Dense(64, activation='relu'))

model.add(Dense(32, activation='relu'))

model.add(Dense(1, activation='sigmoid'))

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

model.summary()
```

Model: "sequential_1"

Layer (type)	Output Shape	Param #
dense_2 (Dense)	(None, 128)	640,128
dense_3 (Dense)	(None, 64)	8,256
dense_4 (Dense)	(None, 32)	2,080
dense_5 (Dense)	(None, 1)	33

Total params: 650,497 (2.48 MB)

Trainable params: 650,497 (2.48 MB)

Non-trainable params: 0 (0.00 B)

training

In [38]:

```
from imblearn.over_sampling import SMOTE

smote = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train_tfidf, y_train)

print("Shape of X_train_resampled:", X_train_resampled.shape)
print("Shape of y_train_resampled:", y_train_resampled.shape)

history = model.fit(X_train_resampled, y_train_resampled, epochs=5, batch_size=32, valid
ation_split=0.2)
```

```
Shape of X_train_resampled: (27238, 5000)
Shape of y_train_resampled: (27238,)
Epoch 1/5
681/681 ━━━━━━━━━━━━━━━━━━━━ 417s 610ms/step - accuracy: 0.9306 - loss: 0.1743 - val_accu
racy: 0.9989 - val_loss: 0.0067
Epoch 2/5
681/681 ━━━━━━━━━━━━━━━━━━━━ 266s 351ms/step - accuracy: 0.9986 - loss: 0.0064 - val_accu
racy: 0.9978 - val_loss: 0.0106
Epoch 3/5
681/681 ━━━━━━━━━━━━━━━━━━━━ 210s 276ms/step - accuracy: 0.9990 - loss: 0.0029 - val_accu
racy: 1.0000 - val_loss: 7.2505e-04
Epoch 4/5
681/681 ━━━━━━━━━━━━━━━━━━━━ 139s 184ms/step - accuracy: 0.9999 - loss: 3.7620e-04 - val_
accuracy: 1.0000 - val_loss: 1.1379e-04
Epoch 5/5
681/681 ━━━━━━━━━━━━━━━━━━━━ 119s 151ms/step - accuracy: 0.9999 - loss: 4.4538e-04 - val_
accuracy: 1.0000 - val_loss: 6.0720e-05
```

evalute model

In [43]:

```
from sklearn.metrics import classification_report, confusion_matrix

# Evaluate the model on the test data
loss, accuracy = model.evaluate(X_test_tfidf, y_test, verbose=0)
print(f'Test Accuracy: {accuracy:.4f}')

# Get predictions
y_pred = (model.predict(X_test_tfidf) > 0.5).astype("int32")

# Display classification report and confusion matrix
print("Classification Report:")
print(classification_report(y_test, y_pred))

print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))
```

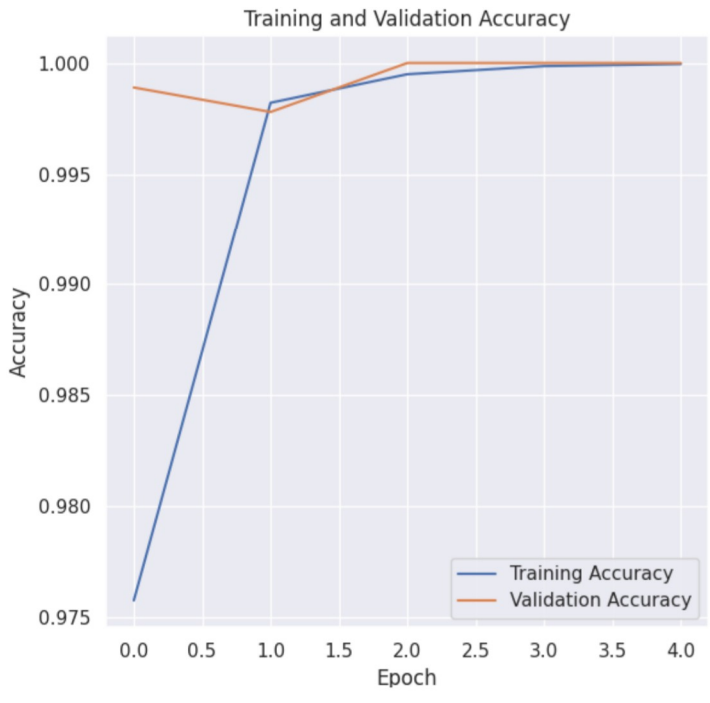
```
Test Accuracy: 0.9852
112/112 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step
Classification Report:
              precision    recall  f1-score   support

           0       0.99      1.00      0.99      3395
           1       0.95      0.75      0.84       181

    accuracy                           0.99      3576
   macro avg       0.97      0.87      0.91      3576
weighted avg       0.98      0.99      0.98      3576

Confusion Matrix:
[[3388    7]
 [  46  135]]
```

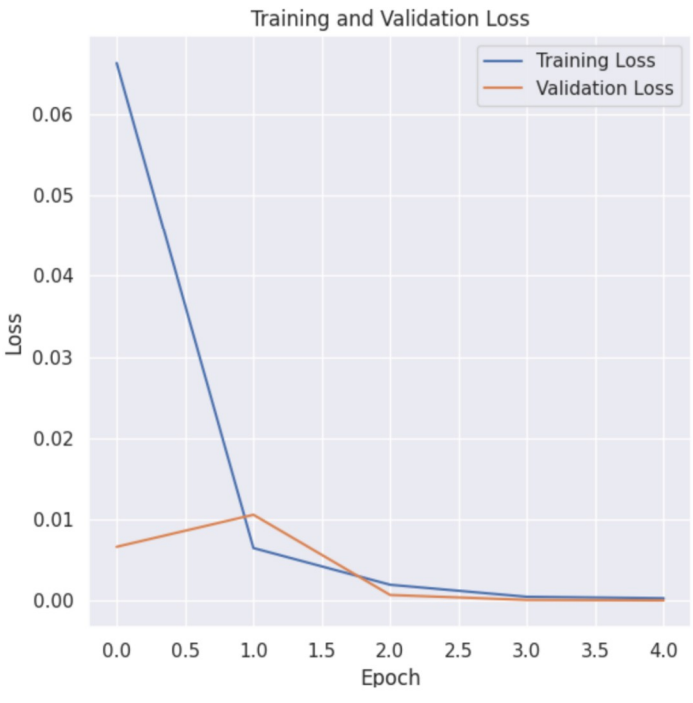
In [44]:

```
from sklearn.metrics import classification_report, confusion_matrix

# Evaluate the model on the test data
```

```
loss, accuracy = model.evaluate(X_test_tfidf, y_test, verbose=0)
print(f'Test Accuracy: {accuracy:.4f}')

# Get predictions
y_pred = (model.predict(X_test_tfidf) > 0.5).astype("int32")

# Display classification report and confusion matrix
print("Classification Report:")
print(classification_report(y_test, y_pred))

print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))

# Plot training and validation accuracy and loss
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()

plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()

plt.tight_layout()
plt.show()
```

```
Test Accuracy: 0.9852
112/112 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step
Classification Report:
              precision    recall  f1-score   support

           0       0.99      1.00      0.99      3395
           1       0.95      0.75      0.84       181

    accuracy                           0.99      3576
   macro avg       0.97      0.87      0.91      3576
weighted avg       0.98      0.99      0.98      3576

Confusion Matrix:
[[3388    7]
 [  46  135]]
```





save model

In [46]:

```
model.save('fraudulent_job_model.keras')
```