Import Libraries

Masukan library yang akan digunakan untuk menganalisa dataset dengan menggunakan metode Deep Learning

In [1]:
```
import warnings
warnings.filterwarnings('ignore')

import pandas as pd
import numpy as np
import wordcloud
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

from tensorflow.keras.models import Model, Sequential
from tensorflow.keras.layers import LSTM, Activation, Dense, Dropout, Input, Embedding
from tensorflow.keras.layers import MaxPooling1D
from tensorflow.keras.optimizers import Adam, RMSprop
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.preprocessing import sequence
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.callbacks import EarlyStopping
%matplotlib inline
```

Load Dataset dengan Google Drive

Sambungkan google drive dengan import drive yang sudah terintregasi dengan Google Colaboratory. Kemudian, ekstraksi dataset format **.csv** dengan menggunakan library Pandas.

In [2]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount
("/content/drive", force_remount=True).
```

In [3]:
```
spam_data = pd.read_csv('drive/My Drive/Dataset/spam_dataset.csv', delimiter=',', encodinc
spam_data.head()
```

Out[3]:

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

Check data NaN

Sebelum dianalisa pastikan dataset harus diperiksa dengan **feature list** agar mengetahui apakah data sudah bersih atau masih kotor? Tidak semua, dataset bersih ada yang harus diperhatikan dalam menganalisa suatu data yaitu menghilangkan beberapa isi kolom dan baris dalam data.

In [4]:
```
feat_list = list(spam_data.columns.values)

for feat in feat_list:
  print(feat, ': ',sum(pd.isnull(spam_data[feat])))
```

```
v1 :  0
v2 :  0
Unnamed: 2 :  5522
Unnamed: 3 :  5560
Unnamed: 4 :  5566
```

In [ ]:
```
spam_data.isnull().any().sum()
```

Out[ ]:
```
3
```

Jumlah data NaN yang diketahui ada 3 label. Maka dari itu data NaN harus dihapus agar tidak terjadi noise pada dataset.

Menghapus data NaN

Setelah mengatahui dataset yang berisikan **NaN** atau **Not a Number**. Maka, hilangkan data yang berisikan NaN tersebut agar memudahkan dalam menganalisa suatu data. Setelah menghapus data NaN, tahapan selanjutnya mengubah nama dataset tersebut berdasarkan kolom yang awalnya **v1** menjadi **label** dan **v2** menjadi **text**.

In [5]:
```
spam_data.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], axis=1, inplace=True)
spam_data = spam_data.rename(columns={"v1":"label", "v2":"text"})
spam_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   label   5572 non-null   object
 1   text    5572 non-null   object
dtypes: object(2)
memory usage: 87.2+ KB
```

Distribusi Target Data Variabel

Dalam dataset memiliki 4825 pada data pesan **Ham** dan 747 pada data pesan **Spam**.

In [6]:
```
spam_data.describe()
```

Out[6]:

	label	text
count	5572	5572
unique	2	5169
top	ham	Sorry, I'll call later
freq	4825	30

In [7]:
```
spam_data.groupby('label').describe()
```

Out[7]:

	text			
	count	**unique**	**top**	**freq**
label				

	text			
	count	unique	top	freq
label				
ham	4825	4516	Sorry, I'll call later	30
spam	747	653	Please call our customer service representativ...	4

Mengetahui Class Dataset

Dalam dataset ini, data SMS Spam harus menggunakan label class agar memudahkan dalam menganalisis suatu data. Ada berapa class dalam tiap data?

In [8]:
```
spam_data.label.value_counts()
```

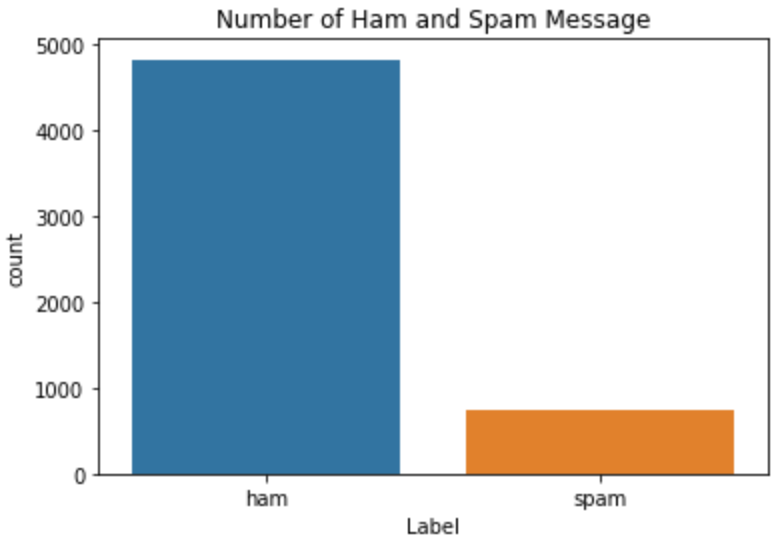
Out[8]:
```
ham     4825
spam     747
Name: label, dtype: int64
```

Visualisasi Data

Setelah data dianalisa untuk mengetahui jumlah data yang akan dianalisis. Maka, lakukan visualisasi agar memudahkan dalam analisis data. Visualisasi yang dipakai menggunakan plot Bar yang terdapat dalam library **Seaborn** dan **Matplotlib**.

In [9]:
```
sns.countplot(spam_data.label)
plt.xlabel('Label')
plt.title('Number of Ham and Spam Message')
```

Out[9]:
```
Text(0.5, 1.0, 'Number of Ham and Spam Message')
```



Visualisasi diatas membuktikan bahwa label Ham memiliki nilai yang sangat tinggi dibandingkan dengan spam.

Menambahkan Label Numberik pada Spam

Target data harus dalam bentuk numerik untuk model klasifikasi menggunakan metode Deep Learning.

In [10]:
```
spam_data['spam'] = spam_data['label'].map( {'spam': 1, 'ham': 0} ).astype(int)
spam_data.head(20)
```

Out[10]:

	label	text	spam
0	ham	Go until jurong point, crazy.. Available only ...	0
1	ham	Ok lar... Joking wif u oni...	0
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	1
3	ham	U dun say so early hor... U c already then say...	0
4	ham	Nah I don't think he goes to usf, he lives aro...	0
5	spam	FreeMsg Hey there darling it's been 3 week's n...	1
6	ham	Even my brother is not like to speak with me. ...	0
7	ham	As per your request 'Melle Melle (Oru Minnamin...	0
8	spam	WINNER!! As a valued network customer you have...	1
9	spam	Had your mobile 11 months or more? U R entitle...	1
10	ham	I'm gonna be home soon and i don't want to tal...	0
11	spam	SIX chances to win CASH! From 100 to 20,000 po...	1
12	spam	URGENT! You have won a 1 week FREE membership ...	1
13	ham	I've been searching for the right words to tha...	0
14	ham	I HAVE A DATE ON SUNDAY WITH WILL!!	0
15	spam	XXXMobileMovieClub: To use your credit, click ...	1
16	ham	Oh k...i'm watching here:)	0
17	ham	Eh u remember how 2 spell his name... Yes i di...	0
18	ham	Fine if thatåÕs the way u feel. thatåÕs the wa...	0
19	spam	England v Macedonia - dont miss the goals/team...	1

In [11]:
```
spam_data['length'] = spam_data['text'].apply(len)
spam_data.head(10)
```

Out[11]:

	label	text	spam	length
0	ham	Go until jurong point, crazy.. Available only ...	0	111
1	ham	Ok lar... Joking wif u oni...	0	29
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	1	155
3	ham	U dun say so early hor... U c already then say...	0	49
4	ham	Nah I don't think he goes to usf, he lives aro...	0	61
5	spam	FreeMsg Hey there darling it's been 3 week's n...	1	148
6	ham	Even my brother is not like to speak with me. ...	0	77
7	ham	As per your request 'Melle Melle (Oru Minnamin...	0	160
8	spam	WINNER!! As a valued network customer you have...	1	158
9	spam	Had your mobile 11 months or more? U R entitle...	1	154

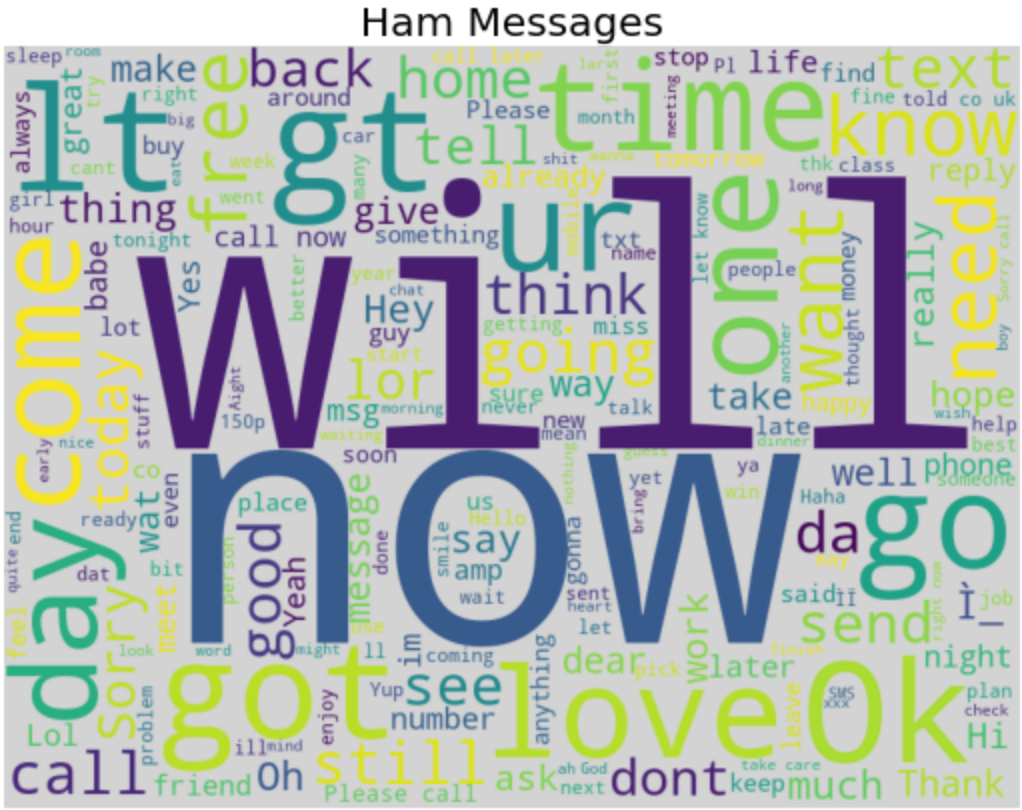
Gunakan Visualisasi WordCloud

Dalam kasus NLP atau Natural Language Processing, Wordcloud sangat membantu user untuk mengetahui kata-kata apa saja yang paling banyak dalam dataset tersebut. Apakah data Ham lebih banyak dibandingkan dengan Spam atau sebaliknya?

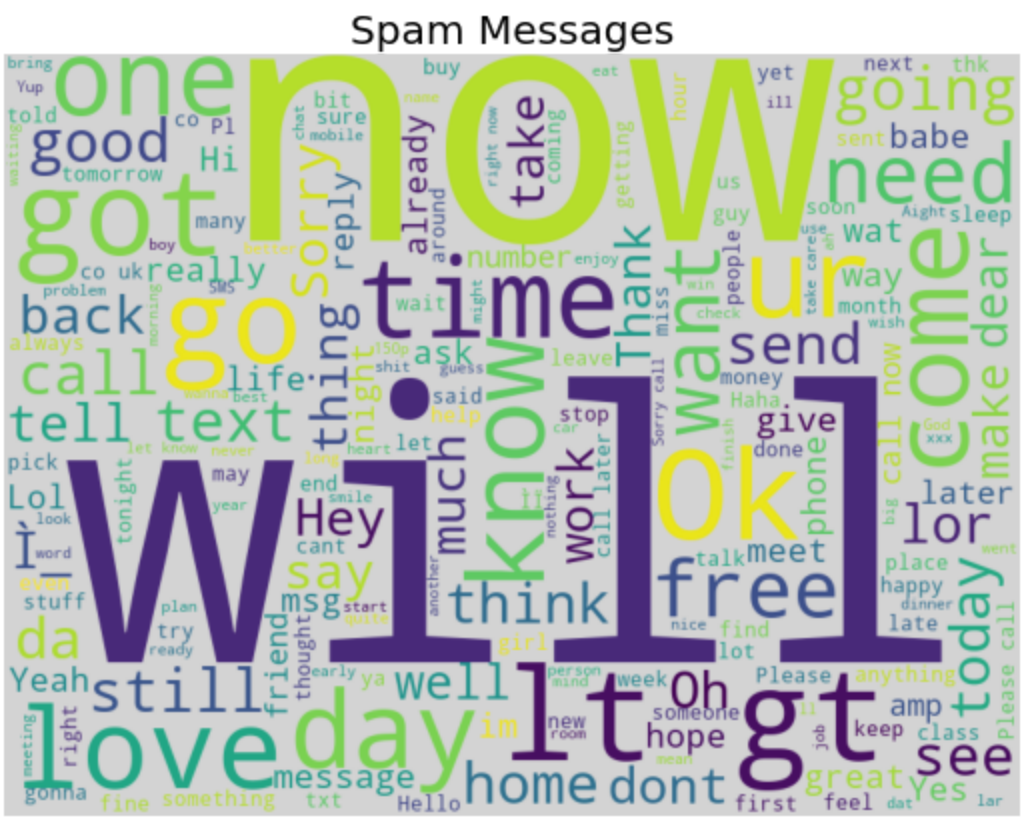
In [12]:
```
data_ham  = spam_data[spam_data['spam'] == 0].copy()
data_spam = spam_data[spam_data['spam'] == 1].copy()
```

In [13]:
```
def show_wordcloud(data_spam_or_ham, title):
    text = ' '.join(data_spam_or_ham['text'].astype(str).tolist())
    stopwords = set(wordcloud.STOPWORDS)

    fig_wordcloud = wordcloud.WordCloud(stopwords=stopwords,background_color='lightgrey',
                    colormap='viridis', width=800, height=600).generate(text)

    plt.figure(figsize=(10,7), frameon=True)
    plt.imshow(fig_wordcloud)
    plt.axis('off')
    plt.title(title, fontsize=20)
    plt.show()
```

In [14]:
```
show_wordcloud(spam_data, "Ham Messages")
```



In [15]:
```
show_wordcloud(spam_data, "Spam Messages")
```



Jika dilihat dari segi visualisasi *wordcloud* Ham dengan Spam. Ada beberapa kata-kata yang hampir sama dalam penyebutan text dan paling besar karena penyebutan kata tersebut sering dipakai.

Setelah mengetahui visualisasi text dengan library **WordCloud**. Maka tambahkan berapa jumlah text data pesan Spam pada pesan yang akan dianalisa.

In [16]:
```
spam_data[spam_data.length == 200].text.iloc[0]
```

Out[16]:
```
'Open rebtel with firefox. When it loads just put plus sign in the user name place, and it will show you two numbers. The lower number is my number. Once you pick that number the pin will display okay!'
```

Text Preprocessing

Pengolahan dasar untuk tugas NLP termasuk dalam konversi teks ke *lowercase* dan menghapuskan punctuation dan **stopwords**. Step yang akan dijalankan, khususnya pada tugas Klasifikasi Teks, adalah:

- **Tokenization**

Ayo mulai untuk analisa pesan text!.

In [17]:
```
vocab_size = 500
oov_token = "<OOV>"
max_length = 250
embedding_dim = 25
```

In [18]:
```
encode = ({'ham': 0, 'spam': 1})
spam_data = spam_data.replace(encode)

spam_data.head()
```

Out[18]:

	label	text	spam	length
0	0	Go until jurong point, crazy.. Available only ...	0	111

	label	text	spam	length
1	0	Ok lar... Joking wif u oni...	0	29
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	1	155
3	0	U dun say so early hor... U c already then say...	0	49
4	0	Nah I don't think he goes to usf, he lives aro...	0	61

In [19]:
```
X = spam_data['text']
y = spam_data['label']

le = LabelEncoder()
y = le.fit_transform(y)
y = y.reshape(-1,1)
```

Tokenizer

Dalam kasus NLP, Tokenizer berguna untuk mengetahui teks yang akan dibaca berdasarkan jumlah kalimat atau kata-kata dalam data pesan tersebut.

In [20]:
```
tokenizer = Tokenizer(num_words=vocab_size, oov_token=oov_token)
tokenizer.fit_on_texts(X)

X = tokenizer.texts_to_sequences(X)
```

In [21]:
```
X = np.array(X)
y = np.array(y)
```

In [22]:
```
X = pad_sequences(X, maxlen=max_length)
```

Split the Data

In [23]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=7)
```

In [24]:
```
print('Data Train : {shape}'.format(shape=X_train.shape))
print('Data Test : {shape}'.format(shape=X_test.shape))
print('Data Train (label) : {shape}'.format(shape=y_train.shape))
print('Data Test (label) : {shape}'.format(shape=y_test.shape))
```

```
Data Train : (4457, 250)
Data Test : (1115, 250)
Data Train (label) : (4457, 1)
Data Test (label) : (1115, 1)
```

In [25]:
```
import keras.callbacks
from timeit import default_timer as timer

class TimingCallback(keras.callbacks.Callback):
    def __init__(self, logs={}):
        self.logs=[]
    def on_epoch_begin(self, epoch, logs={}):
        self.starttime = timer()
    def on_epoch_end(self, epoch, logs={}):
        self.logs.append(timer()-self.starttime)

cb = TimingCallback()
```

Recurrent Neural Network

Setelah tahapan pembagian dataset langkah selanjutnya menggunakan metode RNN dengan layer LSTM (Long Short Time Memory). Kemudian, kompilasi model dengan optimasi menggunakan Adam atau RMSprop dan gunakan loss '**binary_crossentropy**' dikarenakan dataset memiliki 2 kelas.

In [26]:
```
import tensorflow as tf

model = tf.keras.Sequential([
                             tf.keras.layers.Embedding(vocab_size, embedding_dim, input_le
                             tf.keras.layers.MaxPooling1D(pool_size=2),
                             tf.keras.layers.Dense(64, activation='relu'),
                             tf.keras.layers.LSTM(64, dropout=0.4, recurrent_dropout=0.4),
                             tf.keras.layers.Dense(1, activation='sigmoid')
                             ])

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

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 250, 25)           12500
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 125, 25)           0
_________________________________________________________________
dense (Dense)                (None, 125, 64)           1664
_________________________________________________________________
lstm (LSTM)                  (None, 64)                33024
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 65
=================================================================
Total params: 47,253
Trainable params: 47,253
Non-trainable params: 0
_________________________________________________________________
```

In [27]:
```
EPOCHS = 10
BATCH_SIZE = 64
```

In [28]:
```
history = model.fit(X_train, y_train, epochs=EPOCHS, batch_size=BATCH_SIZE,
                    validation_data=(X_test, y_test), validation_split=0.4,
                    callbacks=[EarlyStopping(monitor='val_loss', patience=5, min_delta=0.0

print(cb.logs)
```

```
Epoch 1/10
42/42 [==============================] - 10s 243ms/step - loss: 0.4541 - accuracy: 0.8657
- val_loss: 0.3759 - val_accuracy: 0.8559
Epoch 2/10
42/42 [==============================] - 10s 236ms/step - loss: 0.2512 - accuracy: 0.8994
- val_loss: 0.1787 - val_accuracy: 0.9613
Epoch 3/10
42/42 [==============================] - 10s 233ms/step - loss: 0.1060 - accuracy: 0.9686
- val_loss: 0.0997 - val_accuracy: 0.9675
Epoch 4/10
42/42 [==============================] - 10s 232ms/step - loss: 0.0608 - accuracy: 0.9806
- val_loss: 0.0847 - val_accuracy: 0.9725
Epoch 5/10
42/42 [==============================] - 10s 232ms/step - loss: 0.0465 - accuracy: 0.9847
- val_loss: 0.0753 - val_accuracy: 0.9764
Epoch 6/10
```

```
42/42 [==============================] - 10s 231ms/step - loss: 0.0311 - accuracy: 0.9910
- val_loss: 0.0708 - val_accuracy: 0.9809
Epoch 7/10
42/42 [==============================] - 10s 235ms/step - loss: 0.0228 - accuracy: 0.9936
- val_loss: 0.0675 - val_accuracy: 0.9809
Epoch 8/10
42/42 [==============================] - 10s 234ms/step - loss: 0.0231 - accuracy: 0.9933
- val_loss: 0.0662 - val_accuracy: 0.9826
Epoch 9/10
42/42 [==============================] - 10s 231ms/step - loss: 0.0113 - accuracy: 0.9974
- val_loss: 0.0824 - val_accuracy: 0.9815
Epoch 10/10
42/42 [==============================] - 10s 233ms/step - loss: 0.0080 - accuracy: 0.9974
- val_loss: 0.0806 - val_accuracy: 0.9781
[13.293889764000028, 10.158393040000078, 10.006381653999938, 9.967311012999971, 9.96830600
2999952, 9.92436397099982, 10.075143024999988, 10.07819435700003, 9.93481187700013, 10.027
499567000177]
```

In [29]:
```
print(sum(cb.logs))
```

```
103.43429427100011
```

Evaluasi Model

Setelah menjalankan model fitting. Selanjutnya evaluasi model untuk melihat hasil akurasi dari model RNN tersebut.

In [30]:
```
result = model.evaluate(X_test, y_test)
print('Test set')

loss = result[0]
accuracy = result[1]

print(f'Loss: {loss*100:.2f}%')
print(f'Accuracy: {accuracy*100:.2f}%')
```

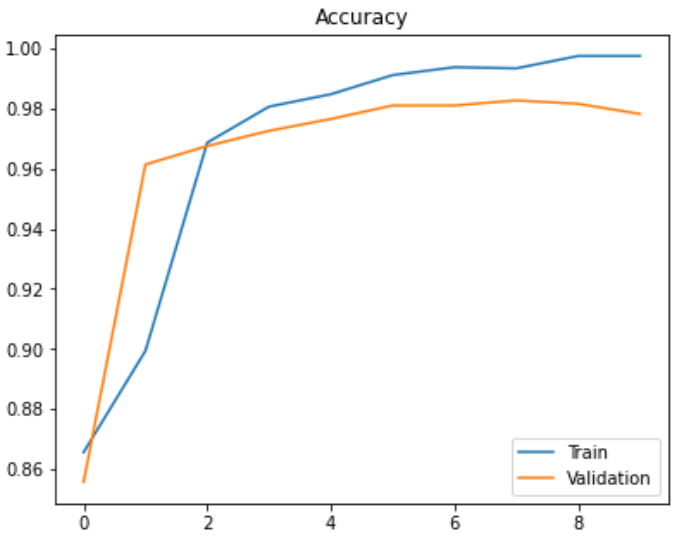
```
35/35 [==============================] - 1s 19ms/step - loss: 0.0626 - accuracy: 0.9839
Test set
Loss: 6.26%
Accuracy: 98.39%
```

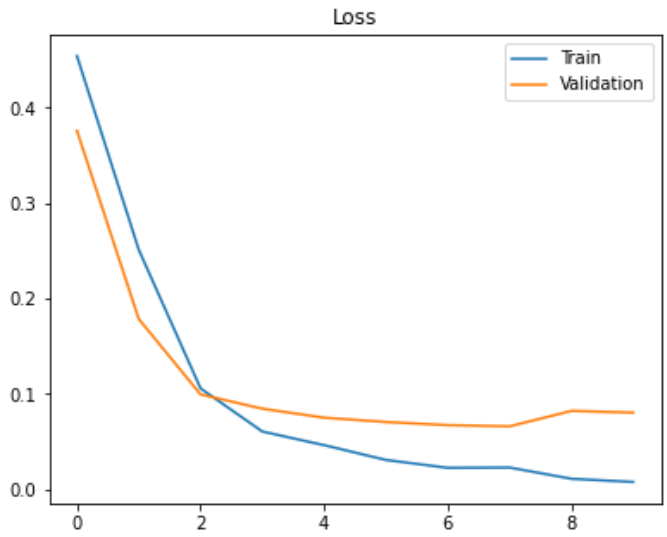
Visualisasi Plot Grafik Model

Setelah mengetahui hasil akurasi dan loss. Langkah selanjutnya adalah visualisasikan plot grafik apakah terjadi underfitting atau overfitting pada model tersebut.

In [31]:
```
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

loss = history.history['loss']
val_loss = history.history['val_loss']

epochs_range = range(EPOCHS)

plt.figure(figsize=(14, 5))
plt.subplot(1, 2, 1)
plt.plot(epochs_range, acc, label='Train')
plt.plot(epochs_range, val_acc, label='Validation')
plt.legend(loc='lower right')
plt.title('Accuracy')

plt.subplot(1, 2, 2)
```

```
plt.plot(epochs_range, loss, label='Train')
plt.plot(epochs_range, val_loss, label='Validation')
plt.legend(loc='upper right')
plt.title('Loss')
plt.show()
```





Prediksi Model Data

Setelah model berhasil dijalankan dan akurasi mendukung. Maka, gunakan prediksi teks apakah pesan tersebut mengandung spam atau tidak?

In [32]:
```
def get_predictions(texts):
    texts = tokenizer.texts_to_sequences(texts)
    texts = sequence.pad_sequences(texts, maxlen=max_length)
    preds = model.predict(texts)
    if(preds[0] > 0.5):
        print("SPAM MESSAGE")
    else:
        print('NOT SPAM')
```

In [33]:
```
# Spam Message
texts=["Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005"]
get_predictions(texts)
```

```
SPAM MESSAGE
```

In [34]:
```
#Not Spam Message
texts = ["Hi man, I was wondering if we can meet tomorrow."]
get_predictions(texts)
```

```
NOT SPAM
```