# Import Data

In [1]:
```
!pip install -r '../requirements.txt'

import nltk

nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')

path_to_csv = '/Users/macbookair/Documents/SEM 3/NLP/data/dataset_m
```

```
ERROR: Could not open requirements file: [Errno 2] No such file or
directory: '../requirements.txt'

[nltk_data] Downloading package punkt to
[nltk_data]     /Users/macbookair/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /Users/macbookair/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```

In [2]:
```
import re
import random
import pandas as pd
import numpy as np
from tensorflow.keras.utils import plot_model
from keras.models import load_model
import matplotlib.pyplot as plt

pd.set_option('mode.chained_assignment', None)
```

In [3]:
```
data=pd.read_csv(path_to_csv, nrows=20)
```

In [4]:
```
data.head()
```

Out[4]:

| | Question_ID | Questions | Jawaban |
|---|---|---|---|
| **0** | 1590140 | Apa yang dimaksud dengan penyakit mental? | Penyakit mental adalah kondisi kesehatan yang ... |
| **1** | 2110618 | Siapa yang terpengaruh oleh penyakit mental? | Diperkirakan bahwa penyakit mental mempengaruh... |
| **2** | 6361820 | Apa penyebab penyakit mental? | Diperkirakan bahwa penyakit mental mempengaruh... |
| **3** | 9434130 | Apa sajakah tanda-tanda peringatan penyakit me... | Gejala gangguan kesehatan mental bervariasi te... |
| **4** | 7657263 | Apakah penderita penyakit jiwa bisa sembuh? | Ketika penyembuhan dari penyakit mental, ident... |

# Data preprocessing

In [5]:
```
def cleaning(text):
    text = re.sub(r'@[A-Za-a0-9]+',' ',text)
    text = re.sub(r'#[A-Za-z0-9]+',' ',text)
    text = re.sub(r"http\S+",' ',text)
    text = re.sub(r'[0-9]+',' ',text)
    text = re.sub(r"[-()\"#/@;:<>{}'+=~|.,_]", '', text)
    text = re.sub(r"(?:\@|https?\://)\S+", '', text)
    text = re.sub(r'[^\x00-\x7f]', '', text)
    text = re.sub(r'\n', ' ', text)
    text = text.strip(' ')
    text = text.lower()
    return text

data['Questions'] = data['Questions'].apply(cleaning)
data['Jawaban'] = data['Jawaban'].apply(cleaning)
```

In [6]:
```
pasangan=[]

for i in range(data.shape[0]):
  pasangan.append(((data['Questions'][i]),data['Jawaban'][i]))
```

In [7]:
```
pasangan
```

```
ampaknya memiliki gejala gangguan jiwa?',
  'meskipun situs web ini tidak dapat menggantikan nasihat profesi
onal kami mendorong mereka yang memiliki gejala untuk berbicara de
ngan teman dan anggota keluarga mereka dan mencari nasihat seorang
profesional kesehatan mental semakin cepat kondisi kesehatan menta
l diidentifikasi dan dirawat semakin cepat mereka bisa berada di j
alan menuju pemulihan jika anda mengenal seseorang yang mengalami
masalah jangan berasumsi bahwa masalah itu akan sembuh sendiri bia
rkan mereka tahu bahwa anda peduli pada mereka dan ada pilihan per
awatan yang tersedia yang akan membantu mereka sembuh bicaralah de
ngan seorang profesional atau penasihat kesehatan mental jika anda
pikir teman atau anggota keluarga anda mengalami gejala kondisi ke
sehatan mental jika orang yang terkasih tahu bahwa anda mendukung
mereka mereka akan lebih cenderung mencari bantuan'),
 ('bagaimana saya bisa menemukan ahli kesehatan mental untuk diri
saya sendiri atau anak saya?',
  'merasa nyaman dengan profesional yang anda atau anak anda beker
ja sama sangat penting untuk keberhasilan perawatan menemukan prof
esional yang paling sesuai dengan kebutuhan anda mungkin memerluka
n penelitian mulailah dengan mencari penyedia di daerah anda'),
```

In [8]:
```
pertanyaan = []
jawaban = []
input_tokens = set()
target_tokens = set()

for line in pasangan:
  tanya, jawab = line[0], line[1]
  pertanyaan.append(tanya)

  jawab = " ".join(re.findall(r"[\w']+|[^\s\w]", jawab))
  jawab = '<START> ' + jawab + ' <END>'
  jawaban.append(jawab)

  for token in re.findall(r"[\w']+|[^\s\w]", tanya):
    if token not in input_tokens:
      input_tokens.add(token)
  for token in jawab.split():
    if token not in target_tokens:
      target_tokens.add(token)

input_tokens = sorted(list(input_tokens))
target_tokens = sorted(list(target_tokens))
num_encoder_tokens = len(input_tokens)
num_decoder_tokens = len(target_tokens)
```

In [9]:
```
pertanyaan
```

Out[9]:
```
['apa yang dimaksud dengan penyakit mental?',
 'siapa yang terpengaruh oleh penyakit mental?',
 'apa penyebab penyakit mental?',
 'apa sajakah tandatanda peringatan penyakit mental?',
 'apakah penderita penyakit jiwa bisa sembuh?',
 'apa yang harus saya lakukan jika saya mengenal seseorang yang ta
mpaknya memiliki gejala gangguan jiwa?',
 'bagaimana saya bisa menemukan ahli kesehatan mental untuk diri s
aya sendiri atau anak saya?',
 'pilihan pengobatan apa yang tersedia?',
 'jika saya terlibat dalam pengobatan apa yang perlu saya ketahu
i?',
 'apa perbedaan antara profesional kesehatan mental?',
 'bagaimana saya dapat menemukan ahli kesehatan mental yang tepat
untuk anak saya atau saya sendiri?',
 'jika saya terlibat dalam pengobatan apa yang perlu saya ketahu
i?',
 'di mana lagi saya bisa mendapatkan bantuan?',
 'apa yang harus saya ketahui sebelum memulai pengobatan baru?',
 'jika saya merasa lebih baik setelah minum obat apakah ini berart
i saya sudah sembuh dan dapat berhenti meminumnya?',
 'bagaimana saya bisa mendapatkan bantuan untuk membayar pengobata
n saya?',
 'ke mana saya bisa mencari terapi',
 'di mana saya dapat mempelajari jenisjenis perawatan kesehatan me
ntal?',
 'apa saja jenisjenis profesional kesehatan mental?',
 'di mana saya bisa mencari kelompok pendukung?']
```

In [10]:
```
jawaban
```

```
an kehidupan sehari hari yang biasa penyakit mental jatuh di sepan
jang rangkaian keparahan beberapa cukup ringan dan hanya mengganggg
u beberapa aspek kehidupan seperti fobia tertentu di ujung lain sp
ektrum terletak penyakit mental yang serius yang mengakibatkan gan
gguan fungsional utama dan gangguan dengan kehidupan sehari hari i
ni termasuk gangguan seperti depresi berat skizofrenia dan ganggua
n bipolar dan mungkin mengharuskan orang tersebut menerima perawat
an di rumah sakit penting untuk mengetahui bahwa penyakit mental a
dalah kondisi medis yang tidak ada hubungannya dengan karakter kec
erdasan atau kemauan seseorang sama seperti diabetes adalah kelain
an pankreas penyakit mental adalah kondisi medis karena biologi ot
ak demikian pula dengan bagaimana seseorang akan mengobati diabete
s dengan obat obatan dan insulin penyakit mental dapat diobati den
gan kombinasi obat dan dukungan sosial perawatan ini sangat efekti
f dengan persen orang yang menerima pengobatan mengalami pengurang
an gejala dan peningkatan kualitas hidup dengan perawatan yang tep
at sangat mungkin bagi orang dengan penyakit mental untuk mandiri
dan sukses <END>',
 '<START> diperkirakan bahwa penyakit mental mempengaruhi dari ora
```

In [11]:
```
input_features_dict = dict([(token, i) for i, token in enumerate(in
target_features_dict = dict([(token, i) for i, token in enumerate(t

reverse_input_features_dict = dict((i, token) for token, i in input
reverse_target_features_dict = dict((i, token) for token, i in targ
```

In [12]:
```
input_features_dict
```

Out[12]:
```
{'?': 0,
 'ahli': 1,
 'anak': 2,
 'antara': 3,
 'apa': 4,
 'apakah': 5,
 'atau': 6,
 'bagaimana': 7,
 'baik': 8,
 'bantuan': 9,
 'baru': 10,
 'berarti': 11,
 'berhenti': 12,
 'bisa': 13,
 'dalam': 14,
 'dan': 15,
 'dapat': 16,
 'dengan': 17,
 'di': 18,
 'dimaksud': 19,
 'diri': 20,
 'gangguan': 21,
 'gejala': 22,
 'harus': 23,
 'ini': 24,
 'jenisjenis': 25,
 'jika': 26,
 'jiwa': 27,
 'ke': 28,
 'kelompok': 29,
 'kesehatan': 30,
 'ketahui': 31,
 'lagi': 32,
 'lakukan': 33,
 'lebih': 34,
 'mana': 35,
 'membayar': 36,
 'memiliki': 37,
 'meminumnya': 38,
 'mempelajari': 39,
 'memulai': 40,
 'mencari': 41,
 'mendapatkan': 42,
 'menemukan': 43,
```

```
'mengenal': 44,
'mental': 45,
'merasa': 46,
'minum': 47,
'obat': 48,
'oleh': 49,
'penderita': 50,
'pendukung': 51,
'pengobatan': 52,
'penyakit': 53,
'penyebab': 54,
'perawatan': 55,
'perbedaan': 56,
'peringatan': 57,
'perlu': 58,
'pilihan': 59,
'profesional': 60,
'saja': 61,
'sajakah': 62,
'saya': 63,
'sebelum': 64,
'sembuh': 65,
'sendiri': 66,
'seseorang': 67,
'setelah': 68,
'siapa': 69,
'sudah': 70,
'tampaknya': 71,
'tandatanda': 72,
'tepat': 73,
'terapi': 74,
'terlibat': 75,
'terpengaruh': 76,
'tersedia': 77,
'untuk': 78,
'yang': 79}
```

# Vektorisasi (Encoder Decoder)

In [13]:
```
max_encoder_seq_length = max([len(re.findall(r"[\w']+|[^\s\w]", tan
max_decoder_seq_length = max([len(re.findall(r"[\w']+|[^\s\w]", jaw

encoder_input_data = np.zeros(
    (len(pertanyaan), max_encoder_seq_length, num_encoder_tokens),
    dtype='float32')
decoder_input_data = np.zeros(
    (len(pertanyaan), max_decoder_seq_length, num_decoder_tokens),
    dtype='float32')
decoder_target_data = np.zeros(
    (len(pertanyaan), max_decoder_seq_length, num_decoder_tokens),
    dtype='float32')
for line, (tanya, jawab) in enumerate(zip(pertanyaan, jawaban)):
    for timestep, token in enumerate(re.findall(r"[\w']+|[^\s\w]",
        encoder_input_data[line, timestep, input_features_dict[toke

    for timestep, token in enumerate(jawab.split()):
        decoder_input_data[line, timestep, target_features_dict[tok
        if timestep > 0:
            decoder_target_data[line, timestep - 1, target_features
```

In [14]:
```
encoder_input_data
```

Out[14]:
```
array([[[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 1.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]],

       [[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 1.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]],

       [[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]],

       ...,

       [[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]],

       [[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]],

       [[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]]], dtype=float32)
```

In [15]:
```
decoder_target_data
```

Out[15]:
```
array([[[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]],

       [[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]],

       [[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]],

       ...,

       [[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]],

       [[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]],

       [[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]]], dtype=float32)
```

# Training Model

In [16]:
```
rom tensorflow import keras
rom keras.layers import Input, LSTM, Dense
rom keras.models import Model
imensionality = 256
atch_size = 10
pochs = 3000

Encoder
ncoder_inputs = Input(shape=(None, num_encoder_tokens))
ncoder_lstm = LSTM(dimensionality, return_state=True)
ncoder_outputs, state_hidden, state_cell = encoder_lstm(encoder_inpu
ncoder_states = [state_hidden, state_cell]

Decoder
ecoder_inputs = Input(shape=(None, num_decoder_tokens))
ecoder_lstm = LSTM(dimensionality, return_sequences=True, return_sta
ecoder_outputs, decoder_state_hidden, decoder_state_cell = decoder_l
ecoder_dense = Dense(num_decoder_tokens, activation='softmax')
ecoder_outputs = decoder_dense(decoder_outputs)
```

In [17]:
```
training_model = Model([encoder_inputs, decoder_inputs], decoder_ou
```

In [18]:
```
training_model.summary()
```

```
Model: "model"
__________________________________________________________________________________________
 Layer (type)                Output Shape              Param #   Connected to
==========================================================================================
 input_1 (InputLayer)        [(None, None, 80)]        0         []

 input_2 (InputLayer)        [(None, None, 562)]       0         []

 lstm (LSTM)                 [(None, 256),             345088    ['input_1[0][0]']
                              (None, 256),
                              (None, 256)]

 lstm_1 (LSTM)               [(None, None, 256),       838656    ['input_2[0][0]',
                              (None, 256),                        'lstm[0][1]',
                              (None, 256)]                        'lstm[0][2]']

 dense (Dense)               (None, None, 562)         144434    ['lstm_1[0][0]']

==========================================================================================
Total params: 1328178 (5.07 MB)
Trainable params: 1328178 (5.07 MB)
Non-trainable params: 0 (0.00 Byte)
__________________________________________________________________________________________
```

In [19]:
```
plot_model(training_model, to_file='model_plot.png', show_shapes=Tr
```

```
You must install pydot (`pip install pydot`) and install graphviz
(see instructions at https://graphviz.gitlab.io/download/) (http
s://graphviz.gitlab.io/download/)) for plot_model to work.
```

In [20]:
```
training_model.compile(optimizer='rmsprop', loss='categorical_cross
history1=training_model.fit([encoder_input_data, decoder_input_data
training_model.save('training_model.h5py')
```

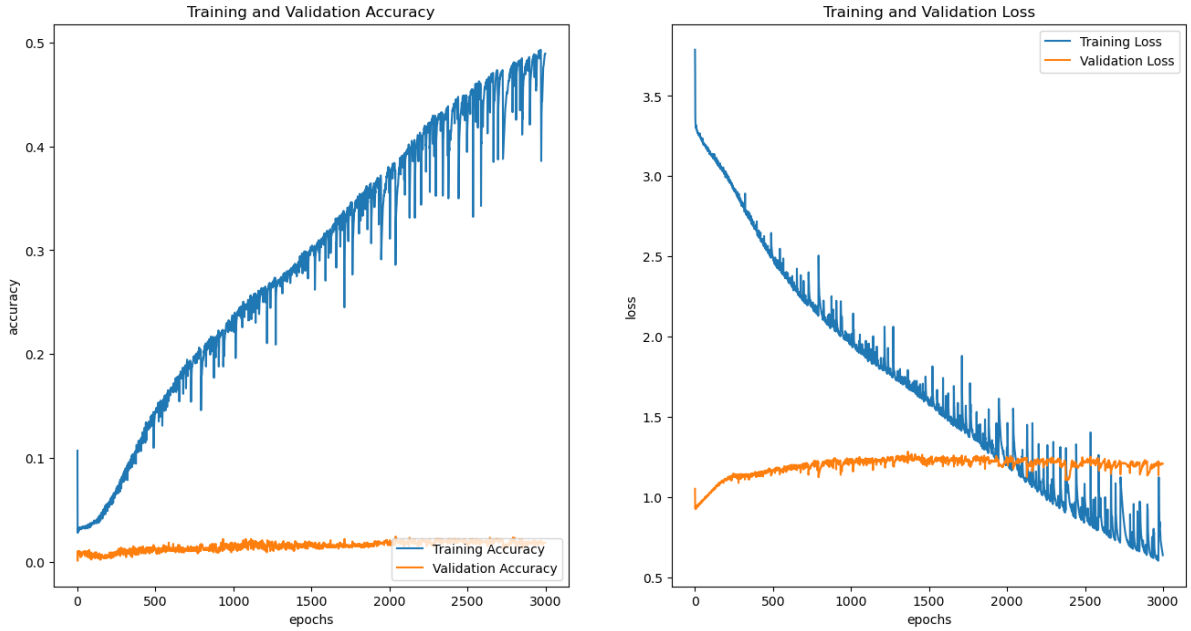
```
Epoch 2993/3000
2/2 [==============================] - 1s 634ms/step - loss: 0.679
8 - accuracy: 0.4831 - val_loss: 1.2093 - val_accuracy: 0.0187
Epoch 2994/3000
2/2 [==============================] - 1s 619ms/step - loss: 0.672
4 - accuracy: 0.4851 - val_loss: 1.2096 - val_accuracy: 0.0187
Epoch 2995/3000
2/2 [==============================] - 1s 656ms/step - loss: 0.666
9 - accuracy: 0.4854 - val_loss: 1.2097 - val_accuracy: 0.0187
Epoch 2996/3000

2/2 [==============================] - 1s 671ms/step - loss: 0.658
6 - accuracy: 0.4883 - val_loss: 1.2088 - val_accuracy: 0.0187
Epoch 2997/3000
2/2 [==============================] - 1s 663ms/step - loss: 0.650
9 - accuracy: 0.4886 - val_loss: 1.2100 - val_accuracy: 0.0187
Epoch 2998/3000
2/2 [==============================] - 1s 599ms/step - loss: 0.646
6 - accuracy: 0.4883 - val_loss: 1.2103 - val_accuracy: 0.0187
Epoch 2999/3000
```

In [21]:
```
acc = history1.history['accuracy']
val_acc = history1.history['val_accuracy']
loss=history1.history['loss']
val_loss=history1.history['val_loss']

plt.figure(figsize=(16,8))
plt.subplot(1, 2, 1)
plt.plot(acc, label='Training Accuracy')
plt.plot(val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')
plt.xlabel("epochs")
plt.ylabel("accuracy")

plt.subplot(1, 2, 2)
plt.plot(loss, label='Training Loss')
plt.plot(val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.xlabel("epochs")
plt.ylabel("loss")
plt.show()
```




# Predictions Model

In [22]:
```
from keras.models import load_model
training_model = load_model('training_model.h5py')
encoder_inputs = training_model.input[0]
encoder_outputs, state_h_enc, state_c_enc = training_model.layers[2
encoder_states = [state_h_enc, state_c_enc]
encoder_model = Model(encoder_inputs, encoder_states)
```

In [23]:
```
latent_dim = 256
decoder_state_input_hidden = Input(shape=(latent_dim,))
decoder_state_input_cell = Input(shape=(latent_dim,))
decoder_states_inputs = [decoder_state_input_hidden, decoder_state_
```

In [24]:
```
decoder_outputs, state_hidden, state_cell = decoder_lstm(decoder_in
decoder_states = [state_hidden, state_cell]
decoder_outputs = decoder_dense(decoder_outputs)
```

In [25]:
```
decoder_model = Model([decoder_inputs] + decoder_states_inputs, [de
```

In [26]:
```
training_model = load_model('training_model.h5py')
encoder_inputs = training_model.input[0]
encoder_outputs, state_h_enc, state_c_enc = training_model.layers[2
encoder_states = [state_h_enc, state_c_enc]
encoder_model = Model(encoder_inputs, encoder_states)

latent_dim = 256
decoder_state_input_hidden = Input(shape=(latent_dim,))
decoder_state_input_cell = Input(shape=(latent_dim,))
decoder_states_inputs = [decoder_state_input_hidden, decoder_state_
decoder_outputs, state_hidden, state_cell = decoder_lstm(decoder_in
decoder_states = [state_hidden, state_cell]
decoder_outputs = decoder_dense(decoder_outputs)
decoder_model = Model([decoder_inputs] + decoder_states_inputs, [de

def decode_response(test_input):
    states_value = encoder_model.predict(test_input)

    target_seq = np.zeros((1, 1, num_decoder_tokens))

    target_seq[0, 0, target_features_dict['<START>']] = 1.

    decoded_sentence = ''

    stop_condition = False
    while not stop_condition:
          output_tokens, hidden_state, cell_state = decoder_model.p

          sampled_token_index = np.argmax(output_tokens[0, -1, :])
          sampled_token = reverse_target_features_dict[sampled_toke
          decoded_sentence += " " + sampled_token

          if (sampled_token == '<END>' or len(decoded_sentence) > m
            stop_condition = True

          target_seq = np.zeros((1, 1, num_decoder_tokens))
          target_seq[0, 0, sampled_token_index] = 1.

          states_value = [hidden_state, cell_state]
    return decoded_sentence
```

In [30]:
```
class ChatBot:
  negative_responses = ("no", "nope", "nah", "naw", "tidak", "nggak
  exit_commands = ("quit", "pause", "exit", "goodbye", "bye", "kelu

  def start_chat(self):
    user_response = input("\nHaloo, aku chatbot pintar!\n")

    if user_response in self.negative_responses:
      print("Ok, bubayy lop u!")
      return
    self.chat(user_response)

  def chat(self, reply):
    while not self.make_exit(reply):
      reply = input(self.generate_response(reply)+"\n")

  def string_to_matrix(self, user_input):
    tokens = re.findall(r"[\w']+|[^\s\w]", user_input)
    user_input_matrix = np.zeros(
      (1, max_encoder_seq_length, num_encoder_tokens),
      dtype='float32')
    for timestep, token in enumerate(tokens):
      if token in input_features_dict:
        user_input_matrix[0, timestep, input_features_dict[token]]
    return user_input_matrix

  def generate_response(self, user_input):
    input_matrix = self.string_to_matrix(user_input)
    chatbot_response = decode_response(input_matrix)
    #Remove <START> and <END> tokens from chatbot_response
    chatbot_response = chatbot_response.replace("<START>",'')
    chatbot_response = chatbot_response.replace("<END>",'')
    return chatbot_response

  def make_exit(self, reply):
    for exit_command in self.exit_commands:
      if exit_command in reply:
        print("Ok, bubayy lop u!")
        return True
    return False

chatbot = ChatBot()
```

# Demo Pertanyaan

In [31]:
```
chatbot.start_chat()
```

```
1/1 [==============================] - 0s 51ms/step
1/1 [==============================] - 0s 70ms/step
1/1 [==============================] - 0s 44ms/step

1/1 [==============================] - 0s 78ms/step
1/1 [==============================] - 0s 45ms/step
1/1 [==============================] - 0s 44ms/step
1/1 [==============================] - 0s 44ms/step
1/1 [==============================] - 0s 53ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 55ms/step
1/1 [==============================] - 0s 45ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 42ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 49ms/step
1/1 [==============================] - 0s 44ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 70ms/step
```