Software Requirements:

- Anaconda: We used it to manage packages required for the project. This platform is mainly used for data science and ML applications. The virtual environment for our project is created here
- Jupyter Notebook: We installed it for our environment and trained and tested our model in this notebook using Python
- PyTorch: This is an open source Machine Learning library used for natural language processing. We used it since it has an optimized tensor library that can be used for deep learning using CPUs
- Simple Transformers: It is based on the Transformers library by Hugging Face. We use the ConvAIModel class to train our chatbot and interact with it
- Tkinter: This is Python library for building a GUI. We used it to build the interface for our chatbot.
- 'Friends' dataset: In order to create the dataset for our default chatbot character('Joey' form 'Friends)' ,we downloaded all the scripts of the show from Kaggle

1. Code for creating the JSON Dataset

```
def load():
  # dictionary of line id to text
  import glob
  list_of_paths=glob.glob("E:/Final Yr Poject/archive/*.txt")
 # print(list_of_paths)
  list_of_paths.sort()



  list_of_dialogues= []
  for ep in list_of_paths:
    path_to_file = ep
    fi = open(path_to_file,encoding="utf8")
    for line in fi:
      l=[]
      stripped_line = line.strip()
      if (stripped_line=="" or stripped_line=="\n"):

        pass


      else:
        name=stripped_line.split()
        if(name[0][-1]==":" or name[0]=="End" or stripped_line== "THE END" ):
          #l.append(stripped_line)
          list_of_dialogues.append(stripped_line)
  #print(list_of_dialogues[:15])
  #print(list_of_dialogues[0])

  s=0
# last_hist = []
#  last_candidates = None
```

```
 u_list=[{'candidates':['my name is joey'],'history':['what is your name?']},{'candidates':['i am an actor'],'history':['what is your profession?']},{'candidates':['i live in new york city'],'history':['where do you live?']},{'candidates':['chandler, ross, rachel, monica and phoebe'],'history':['who is your best friend?']},{'candidates':['yankees'],'history':['which is your favourite baseball team?']}]
 jd = 1
 while (True):

   diag = list_of_dialogues[s]
   if(jd ==1):
     u=dict()
     u['candidates']=[]
     u['history']=[]



   if (diag.split()[0] != "Joey:"):
     jd=0
     #print("hiii")
     p=diag.find(':')
     nj_diag=diag[p+2:].lower().replace(".", " .").replace('!',' !').replace('?',' ?').replace(',',
' ,')
     u['history'].append(nj_diag)

   else:
     jd=1
     po=diag.find(':')
     j_diag=diag[po+2:].lower().replace("."," .").replace('!',' !').replace('?',' ?').replace(',',
' ,')
     u['candidates'].append(j_diag)

     #print(u)
```

```
      u_list.append(u)
    s=s+1




    if (s >= 1000):
      break
  #print(len(list_of_dialogues))
  return u_list

u_list=load()



dataset=[{"personality":['Joey','I am an actor','I am Italian','I love the Yankees','i live in new york','i like sandwiches','my profession is acting','i have seven sisters','chandler is my roommate']
          }]
dataset[0]['utterances']=u_list

import json

json_dataset = json.dumps(dataset)
#print(json_dataset)

with open("new.json", "w") as outfile:
  outfile.write(json_dataset)
```

2. Code for training the model

```
from simpletransformers.conv_ai import ConvAIModel


train_args = {
   "overwrite_output_dir": True,
   "reprocess_input_data": True
}

# Create a ConvAIModel
model = ConvAIModel("gpt", "gpt_personachat_cache", use_cuda=False,
args=train_args)
```

3. Code to fine-tune our model using the JSON dataset

```
model.train_model("data/train.json")
```

4. Code to connect the trained model with the Character Chatbot GUI

```
def get_response(inp,persona):
   if persona == None or persona==[]:
      personality=["i am joey .","i like sandwiches .","i am actor .","i live in new
york .",'i am italian']

   else:
      personality=persona

   history=['hi','hello, how are you','i am fine how abt you']

   response,history= model.interact_single(inp,history,personality=personality)
   print(response)
   return response
```

5. Code for creating the Character Chatbot GUI

```
from tkinter import *




BG_GRAY = "gray19"
BG_COLOR = "gray7"
TEXT_COLOR = "white"

FONT = "Arial 14"
FONT_BOLD = "Arial 13 bold"

class ChatApplication:

    def __init__(self):
        self.window = Tk()
        self.plist=[]
        self.bot_name='Joey'
        self._setup_main_window()

    def run(self):
        self.window.mainloop()

    def _setup_main_window(self):
        self.window.title("Character Chatbot")
        self.window.resizable(width=False, height=False)
        self.window.configure(width=550, height=550, bg=BG_COLOR)

        # head label
        head_label = Label(self.window, bg='gray19', fg='white',
                           text="Welcome to Charcter Chatbot", font=FONT_BOLD, pady=10)
        head_label.place(relwidth=1)
        # tiny divider
        line = Label(self.window, width=450, bg=BG_GRAY)
        line.place(relwidth=1, rely=0.07, relheight=0.012)



        # text widget
        self.text_widget = Text(self.window, width=20, height=2, bg=BG_COLOR,
fg='white',
                                font=FONT, padx=5, pady=5)
        self.text_widget.place(relheight=0.645, relwidth=1, rely=0.08)
        self.text_widget.configure(cursor="arrow", state=DISABLED)

        # scroll bar
        scrollbar = Scrollbar(self.text_widget)
        scrollbar.place(relheight=1, relx=0.974)
        scrollbar.configure(command=self.text_widget.yview)

        # bottom label
        bottom_label = Label(self.window, bg=BG_GRAY, height=150)
        bottom_label.place(relwidth=1, rely=0.730)

        # message entry box
        self.msg_entry = Entry(bottom_label, bg="gray33", fg=TEXT_COLOR,
font=FONT)
        self.msg_entry.place(relwidth=0.74, relheight=0.02, rely=0.030, relx=0.011)
        self.msg_entry.focus()
        self.msg_entry.bind("<Return>", self._on_enter_pressed)

        # send button
        send_button = Button(bottom_label, text="Chat", font=FONT_BOLD, width=20,
bg='gainsboro',command=lambda: self._on_enter_pressed(None))
        send_button.place(relx=0.77, rely=0.030, relheight=0.02, relwidth=0.22)

        #persona message entry box
        self.pmsg_entry = Entry(bottom_label, bg="gray33", fg=TEXT_COLOR, font=FONT)
        self.pmsg_entry.place(relwidth=0.74, relheight=0.02, rely=0.008, relx=0.011)
        self.pmsg_entry.focus()
        self.pmsg_entry.bind("<Return>", self.persona_entry)


        #persone_btn
        persona_btn= Button(bottom_label, text="Character", font=FONT_BOLD, width=15, bg='gainsboro',command=lambda: self.persona_entry(None))
        persona_btn.place(relx=0.77, rely=0.008, relheight=0.02, relwidth=0.22)





    def _on_enter_pressed(self, event):
        msg = self.msg_entry.get()
        self._insert_message(msg, "You")

    def persona_entry(self,event):
        pmsg = self.pmsg_entry.get()


        if pmsg == '':
            self.bot_name='Joey'
        else:
            self.plist= list(map(str,pmsg.split(".")))
            self.bot_name= self.plist[0]
        self.insert_persona(pmsg,"Persona:")




    def insert_persona(self,persona,p):
        self.pmsg_entry.delete(0,END)
        msg0 = f"{'Character'}: {self.plist}\n\n"
        #pstring = msg0[]
        self.text_widget.configure(state=NORMAL)
        self.text_widget.insert(END, msg0)
        self.text_widget.configure(state=DISABLED)

        self.text_widget.see(END)



    def _insert_message(self, msg,  sender):
        if not msg:
            return

        self.msg_entry.delete(0, END)
        msg1 = f"{sender}: {msg}\n\n"
        self.text_widget.configure(state=NORMAL)
        self.text_widget.insert(END, msg1)
        self.text_widget.configure(state=DISABLED)

        msg2 = f"{self.bot_name}: {get_response(msg,self.plist)}\n\n"
        self.text_widget.configure(state=NORMAL)
        self.text_widget.insert(END, msg2)
        self.text_widget.configure(state=DISABLED)

        self.text_widget.see(END)




app = ChatApplication()
app.run()
```