

PROJECT REPORT

(Project Term August-December 2021)

(AI Based Agriculture chatbot for farmers)

Submitted by

(Masapeta Akhileshwar Reddy)

Registration Number: 11904738

(Chimaladenne Dayakar Sainath)

Registration Number : 11912411

Course Code: INT246

Under the Guidance of

(Dr. Sagar pande)

School of Computer Science and Engineering



L OVELY
P ROFESSIONAL
U NIVERSITY

DECLARATION

We hereby declare that the project work entitled “AI Based Agriculture chatbot for farmers” is an authentic record of our own work carried out as requirements of Project for the award of B-Tech degree in computer science from Lovely Professional University, Phagwara, under the guidance of Dr. Sagar Pande, during August to December 2021. All the information furnished in this project report is based on our own intensive work and is genuine.

Name of Student 1: Masapeta Akhileshwar Reddy Registration

Number: 11904738

Name of Student 2: Chimaladenne Dayakar Sainath Registration

Number: 11912411

(Signature of Student 1)

Date:

(Signature of Student 2)

Date:

CERTIFICATE

This is to certify that the declaration statement made by this group of students is correct to the best of my knowledge and belief. They have completed this Project under my guidance and supervision. The present work is the result of their original investigation, effort and study. No part of the work has ever been submitted for any other degree at any University. The Project is fit for the submission and partial fulfillment of the conditions for the award of B. Tech degree in computer science from Lovely Professional University, Phagwara.

Signature and Name of the Mentor

Designation

School of Computer Science and Engineering,

Lovely Professional University, Phagwara, Punjab.

Date:

ACKNOWLEDGEMENT

I would like to express my special thanks of gratitude to Dr. Sagar Pande sir. who gave me the golden opportunity to do this wonderful project on the topic (AI Based Agriculture chatbot for farmers), which also helped me in doing a lot of Research and I came to know about so many new things. I am really thankful to have this opportunity.

Abstract:

Artificial Intelligence and Machine Learning are driving IT industry to new landscape. this chatbot overcomes this problem and provides farmers the better opportunity to obtain the desired information and to scale up with upcoming market trends and technologies in a user-friendly manner. this chatbot is actually an AI chatbot, which is a virtual conversational assistant, through which the users can communicate with the bot as if they are conversing with humans. The focus is on developing the bot in a more intellectual way, that it can even recognize not so well grammatically defined sentences, misspelled words, incomplete phrases, etc. This can help people to converse easily with the bot, since this system uses the Natural Language Processing technique to parse the user queries, identify the key words, match them with Knowledge Base and respond with the accurate results. To make the responses more understandable, the responses are generated using classification algorithms and produce textual responses so that it can be easily perceived by the users.

Introduction:

Agriculture is the major provider of employment to people in many parts of the world. Many people depend on agriculture for their livelihood. Most countries depend on agriculture for their GDP growth. The technology in the field of agriculture is developing day-by-day. Also, a large number of software is being simultaneously developed, to educate the farmers with this technological information. Most of them provide static information about farming, they require large

number of searching steps to get the accurate information and they don't provide an interactive way of querying and response. This system overcomes the above-mentioned drawbacks by providing a user interface, where farmers or any other users can interact effectively to get the desired responses with few numbers of steps. This chatbot, which is a virtual assistant that enable users to get their queries clarified in a user-friendly manner. The

input is obtained from the user, the textual query will undergo pre-processing steps in order to find the category of the query it belongs to, and provide the corresponding response. If the query is based on prediction, then the future predictions on the requested agricultural products will be represented in the graphical format and displayed to the user. The system would educate the new age farmers regarding agriculture information using interactive querying technique. It also helps the farmers to plan their future activities by predicting the future cost of agriculture products.

This project is focusing on creating a simple chat-bot to be used by farmers to get information about the agriculture without visiting the government website or office. A chat-bot is a program which can do real conversations with textual method. Using Artificial Intelligence (AI), chat-bots can simulate human conversations. There are two categories of chat-bots. One category is command-based chat-bots where chat-bots rely on a databank of replies. The user must be very specific while asking the questions so that the bot can answer. Hence, these bots can answer limited set of questions and cannot perform function outside of the code. The other category is chat-bots based on AI or machine learning algorithms; these bots can answer ambiguous questions which means the user do not have to be specific while asking questions. Thus, these bots create replies for the user's queries using Natural Language Processing (NLP). Making an AI based chat-bot is a tedious job and is difficult to understand the process as compared to a databank-based chat-bot. Our project develops an NLP chat-bot as a part of learning to create basic chat-bots. It focuses on giving specific information about the agriculture. Whenever a user asks any query, the bot will first analyse the request, then identifies intents and entities, builds a response and sends it back to the user. Now, intents mean intention of the query and entity means details of that query. For example, if a farmer wants to know the cost of a seeds, then the intent will be seeds and entity will be name of the seed in this case and give the response on seeds.

Motivation:

Chat-bots are motivated by the need of traditional websites to provide a chat facility where a bot is required to be able to chat with user and solve queries. When live agent can handle only two to three operations at a time, chat-bots can operate without an upper limit which really scales up the operations. Also, if any agriculture agent is receiving lots of queries, having a chat-bot on a website takes off the load from support team. Having a chat-bot clearly improves the response rate compared to human support team. In addition, since millennial prefer live chats over a phone call, they find a chat-bot, which provide a highly interactive marketing platform, very attractive. Furthermore, a chat-bot can automate the repetitive tasks. There can be some scenarios where a government receives same queries in a day for many times and support team must respond to each query repetitively. Lastly, the most important advantage of having a chat-bot is that it is available 24/7. No matter what time it is, a user can get a query solved. All these advantages of a chat-bot constitute the motivation to implement an agriculture Chat-bot.

Before implementing agriculture Chat-bot, various existing chat-bots were reviewed such as Amazon Shopping App, YONO SBI chat-bot, Alexa, etc. In order to understand the requirement of a chat-bot, consider an example of Amazon Shopping App. In this app, when a customer buys an item, he/she does not have any information about how to return the item. To get this information, the customer must call and wait to talk to customer representative for a long time. However, this whole process is tedious for a customer. Hence, Amazon created a chat-bot to answer simple queries of customers. Similarly, the agriculture Chat-bot is designed to help farmers to get information on a fingertip.

Objectives:

- With the help of Chat-Bot a farmers need not have to be physically present in government office for the enquiry and thus gaps the bridge between officials and the farmers
- Helping farmers to get basic information about the agriculture
- Creating an easy-to-use chat-bot for farmers which requires user to enter any option provided by the chat-bot.
- farmers willing for information in the agriculture field can have quick information about the agriculture using this chat-bot.

Limitations of existing system:

- Time consuming and hectic
- Needs personal attendance
- Isn't available 24/7

The existing system is not only tedious but also makes a gap between officials and farmers. Surfing the website for small information can be irritating sometimes. It is rather better to just enter your query and get information on your fingertip. Now-a -days there are many changes that occurred in the agriculture system with help of advanced technological improvements. Everything is happening over the internet without any difficulty. In previous days for submitting a small application we had to visit that place, but as the days are passing away it's completely changing. Collecting the applications manually will be hectic procedure and it also needs a lot of manpower. For reducing that manpower and such difficulties many

devices or systems were emerged day by day. One of them is a chat-bot which serves as an efficient communication tool between system and users. Chat-bot bridges the gap between traditional enquiry means and users that are particular about time.

Implemented Technique:

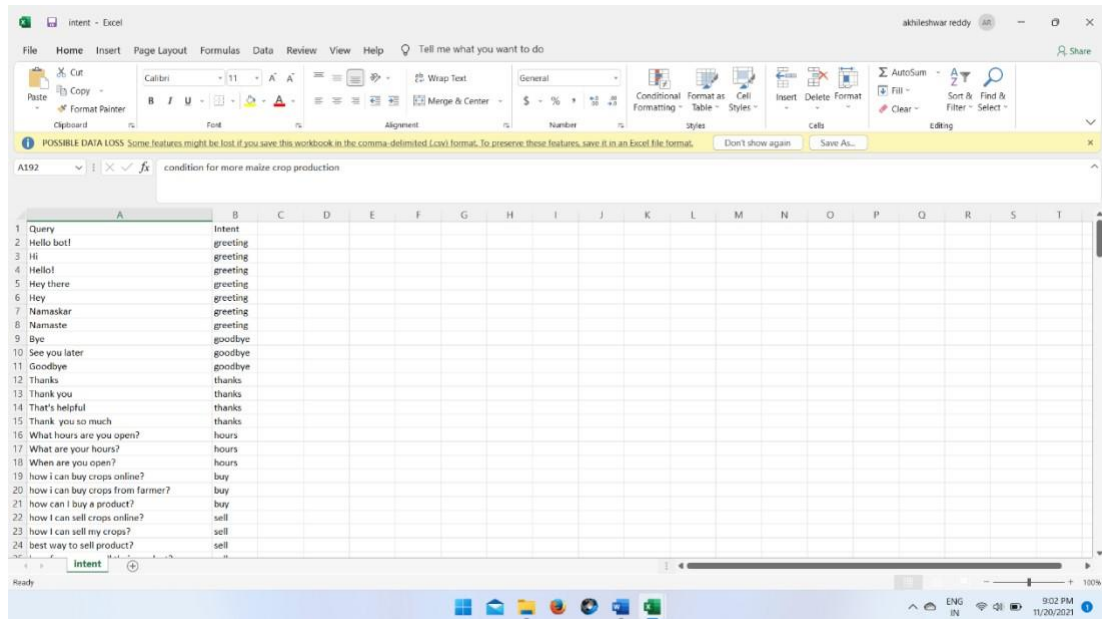
An agriculture chat-bot project is built using artificial algorithms that analyses user's queries and understand user's message. This System is a web application which provides answer to the query of the farmers. farmers just have to query through the bot which is used for chatting. farmers can chat just like we do on normal messaging without keeping in mind about case sensitivity. The System uses built in artificial intelligence to answer the query. If the answer found to be invalid, the admin will correct and update the incorrect answer. Admin can view and update the information at any time. System allows admin to delete the invalid answer or to add a specific answer of that equivalent question.

The User can query any agriculture related activities through the system. The user does not have to personally go anywhere for enquiry. The System analyses the question and then answers to the user. The system answers to the query as if it is answered by the person. With the help of artificial intelligence, the system answers the query asked by the farmers. The system replies using an effective Graphical user interface which implies that as if a real person is talking to the user. The user can query about the agriculture related activities through online with the help of this web application. This system helps the farmers to be updated about the agriculture activities.

Architecture/ Framework:

Chat bots typically provide a text-based user interface, allowing the user to type commands and receive text response. Chat bots are usually stateful services, remembering previous commands in order to provide functionality. When chat bot technology is integrated with popular web services it can be utilized securely by an even larger audience. The college enquiry chat bot will be built using artificial algorithms that analyses user's queries and understand user's message. This System will be a gui application which provides answer to the query of the farmer very effectively. farmer just have to put their query to the bot which is used for chatting. The system will use the artificial intelligence algorithms to give appropriate answers to the user. If the answer is found invalid, then some system to declare the answer as invalid can be incorporated. These invalid answers can be deleted or modified by the system. The farmer will not have to go to the agriculture office for enquiring something. farmer can use the chat bot to get the answers to their queries. farmer can use this web-based system for making enquiries at any point of time. This system may help farmers to stay updated with the agriculture activities. inputs the query in the user interface in the form of text. The user interface receives the user queries and then forwards it to the chatbot application. In the chatbot application, the textual query undergoes a pre-processing stage. Pre-processing steps include Tokenization where the query sentence is tokenized into words, then the stop words are removed, and then the words are stemmed to their root words. If the query is classification based, it would undergo classification using the Navies Bayes classifier, which uses the knowledge base to retrieve the relevant responses.

Data sets:



The screenshot shows an Excel spreadsheet with the following data:

Query	intent
Hello bot!	greeting
Hi	greeting
Hello!	greeting
Hey there	greeting
Hey	greeting
Namaskar	greeting
Namaste	greeting
Bye	goodbye
See you later	goodbye
Goodbye	goodbye
Thanks	thanks
Thank you	thanks
That's helpful	thanks
Thank you so much	thanks
What hours are you open?	hours
What are your hours?	hours
When are you open?	hours
how i can buy crops online?	buy
how i can buy crops from farmer?	buy
how can i buy a product?	buy
how i can sell crops online?	sell
how i can sell my crops?	sell
best way to sell product?	sell

The Dataset contains following columns

- Query
- intent

Code:

```
In [3]: 1 import pandas as pd
2 import numpy as np
3 import pickle as pk
4 from sklearn.feature_extraction.text import CountVectorizer
5 from sklearn.preprocessing import LabelEncoder
6 from sklearn.metrics import confusion_matrix
7 from sklearn.naive_bayes import GaussianNB
8 import re
9 from nltk.stem.porter import PorterStemmer
10 from sklearn.model_selection import train_test_split
11 import keras
12 from keras.models import Sequential
13 from keras.layers import Dense
14 from keras.utils import np_utils
15 from keras.models import load_model
16 import tensorflow
17
18
19
20
21 def trainIntentModel():
22     # Load the dataset and prepare it to the train the model
23
24     # Importing dataset and splitting into words and labels
25     dataset = pd.read_csv('datasets/intent.csv', names=["Query", "Intent"])
26
27     X = dataset["Query"]
28     y = dataset["Intent"]
29
30     unique_intent_list = list(set(y))
31
32     print("Intent Dataset successfully loaded!")
33
```

```

33
34     # Clean and prepare the intents corpus
35     queryCorpus = []
36     ps = PorterStemmer()
37
38     for query in X:
39         query = re.sub('[^a-zA-Z]', ' ', query)
40
41         # Tokenize sentence
42         query = query.split(' ')
43
44         # Lemmatizing
45         tokenized_query = [ps.stem(word.lower()) for word in query]
46
47         # Recreate the sentence from tokens
48         tokenized_query = ' '.join(tokenized_query)
49
50         # Add to corpus
51         queryCorpus.append(tokenized_query)
52
53     # print(queryCorpus)
54     print("Corpus created")
55
56     countVectorizer = CountVectorizer(max_features=800)
57     corpus = countVectorizer.fit_transform(queryCorpus).toarray()
58     print(corpus.shape)
59     print("Bag of words created!")
60

```

```

61     # Save the CountVectorizer
62     pk.dump(countVectorizer, open("saved_state/IntentCountVectorizer.sav", 'wb'))
63     print("Intent CountVectorizer saved!")
64
65     # Encode the intent classes
66     labelencoder_intent = LabelEncoder()
67     y = labelencoder_intent.fit_transform(y)
68     y = np_utils.to_categorical(y)
69     print("Encoded the intent classes!")
70     # print(y)
71
72     # Return a dictionary, mapping labels to their integer values
73     res = {}
74     for cl in labelencoder_intent.classes_:
75         res.update({cl:labelencoder_intent.transform([cl])[0]})
76
77     intent_label_map = res
78     # print(intent_label_map)
79     print("Intent Label mapping obtained!")
80

```

```

80
81 # Initialising the Aritifcial Neural Network
82 classifier = Sequential()
83
84 # Adding the input layer and the first hidden layer
85 classifier.add(Dense(units = 96, kernel_initializer = 'uniform', activation = 'relu', input_dim = 133))
86
87 # Adding the second hidden layer
88 classifier.add(Dense(units = 96, kernel_initializer = 'uniform', activation = 'relu'))
89
90 # Adding the output layer
91 classifier.add(Dense(units = 35, kernel_initializer = 'uniform', activation = 'softmax'))
92
93 # Compiling the ANN
94 classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])
95
96 # Fitting the ANN to the Training set
97 classifier.fit(corpus, y, batch_size = 10, epochs = 500)
98
99 return classifier, intent_label_map
100

```

```

95
96 # Fitting the ANN to the Training set
97 classifier.fit(corpus, y, batch_size = 10, epochs = 500)
98
99 return classifier, intent_label_map
100
101
102
103
104
105
106 intent_model, intent_label_map = trainIntentModel()
107
108 # Save the Intent model
109 intent_model.save('saved_state/intent_model.h5')
110 print("Intent model saved!")
111
112
113

```

```

113
114 def trainEntityModel():
115     # Importing dataset and splitting into words and labels
116     dataset = pd.read_csv('datasets/data-tags.csv', names=["word", "label"])
117     X = dataset.iloc[:, :-1].values
118     y = dataset.iloc[:, 1].values
119     # print(X)
120     print("Entity Dataset successfully loaded!")
121
122     entityCorpus=[]
123     ps = PorterStemmer()
124
125     # Stem words in X
126     for word in X.astype(str):
127         word = [ps.stem(word[0])]
128         entityCorpus.append(word)
129
130     # print(entityCorpus)
131     X = entityCorpus
132     from numpy import array
133     X = array(X)
134     X = X.reshape(len(X),)

```

```

135
136     # Create a bag of words model for words
137     from sklearn.feature_extraction.text import CountVectorizer
138     cv = CountVectorizer(max_features=1500)
139     # X = cv.fit_transform(X.astype('U')).toarray()
140     X = cv.fit_transform(X).toarray()
141     print("Entity Bag of words created!")
142
143     # Save CountVectorizer state
144     pk.dump(cv, open('saved_state/EntityCountVectorizer.sav', 'wb'))
145     print("Entity CountVectorizer state saved!")
146
147     # Encoding categorical data of labels
148     labelencoder_y = LabelEncoder()
149     y = labelencoder_y.fit_transform(y.astype(str))
150     print("Encoded the entity classes!")
151
152     # Return a dict mapping labels to their integer values
153     res = {}
154     for cl in labelencoder_y.classes_:
155         res.update({cl:labelencoder_y.transform([cl])[0]})
156     entity_label_map = res
157     print("Entity Label mapping obtained!")
158
159     # Fit classifier to dataset
160     classifier = GaussianNB()
161     classifier.fit(X, y)
162     print("Entity Model trained successfully!")
163

```

```

163
164     # Save the entity classifier model
165     pk.dump(classifier, open('saved_state/entity_model.sav', 'wb'))
166     print("Trained entity model saved!")
167
168     return entity_label_map
169
170
171
172
173 # Load Entity model
174 entity_label_map = trainEntityModel()
175
176 loadedEntityCV = pk.load(open('saved_state/EntityCountVectorizer.sav', 'rb'))
177 loadedEntityClassifier = pk.load(open('saved_state/entity_model.sav', 'rb'))
178
179
180

```



```

179
180
181
182 def getEntities(query):
183     query = loadedEntityCV.transform(query).toarray()
184
185     response_tags = loadedEntityClassifier.predict(query)
186
187     entity_list=[]
188     for tag in response_tags:
189         if tag in entity_label_map.values():
190             entity_list.append(list(entity_label_map.keys())[list(entity_label_map.values()).index(tag)])
191
192     return entity_list
193
194
195
196

```

```

197
198 import json
199 import random
200
201 with open('datasets/intents.json') as json_data:
202     intents = json.load(json_data)
203
204 # Load model to predict user result
205 loadedIntentClassifier = load_model('saved_state/intent_model.h5')
206 loaded_intent_cv = pk.load(open('saved_state/IntentCountVectorizer.sav', 'rb'))
207
208 USER_INTENT = ""
209

```

```

210
211 def chatbot_response(msg):
212     query = re.sub('[^a-zA-Z]', ' ', msg)
213     # Tokenize sentence
214     query = query.split(' ')
215
216     # Lemmatizing
217     ps = PorterStemmer()
218     tokenized_query = [ps.stem(word.lower()) for word in query]
219
220     # Recreate the sentence from tokens
221     processed_text = ' '.join(tokenized_query)
222
223     # Transform the query using the CountVectorizer
224     processed_text = loaded_intent_cv.transform([processed_text]).toarray()
225
226     # Make the prediction
227     predicted_intent = loaded_intent_classifier.predict(processed_text)
228     # print(predicted_intent)
229     result = np.argmax(predicted_intent, axis=1)
230
231     for key, value in intent_label_map.items():
232         if value == result[0]:
233             # print(key)
234             USER_INTENT = key
235             break
236
237     for i in intents['intents']:
238         if i['tag'] == USER_INTENT:
239             return random.choice(i['responses'])
240

```

```

236
237     for i in intents['intents']:
238         if i['tag'] == USER_INTENT:
239             return random.choice(i['responses'])
240
241
242     # Extract entities from text
243     entities = get_entities(tokenized_query)
244
245     # Mapping between tokens and entity tags
246     token_entity_map = dict(zip(entities, tokenized_query))
247     # print(token_entity_map)

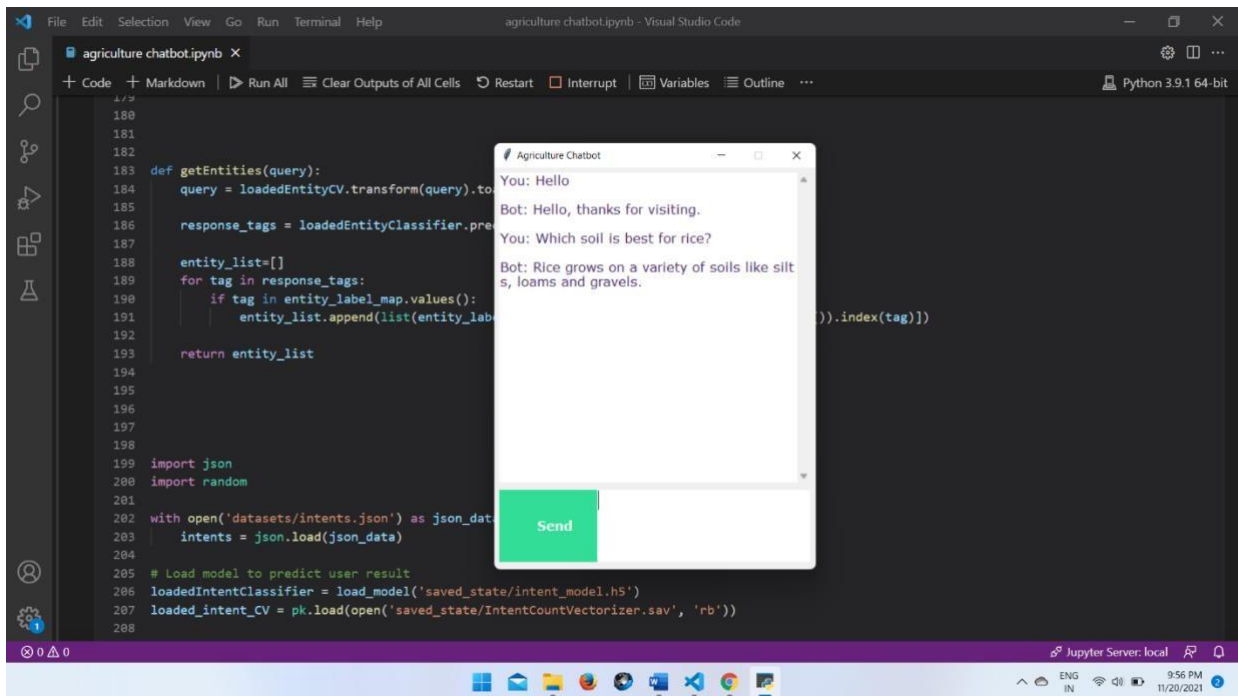
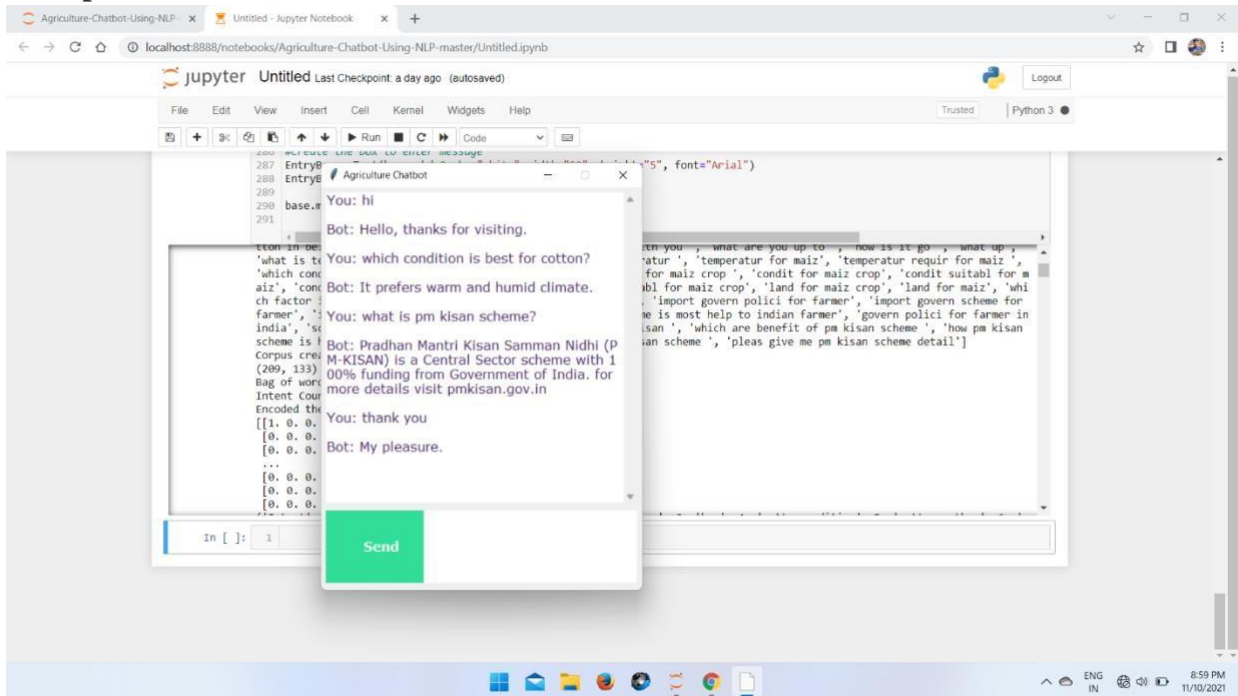
```

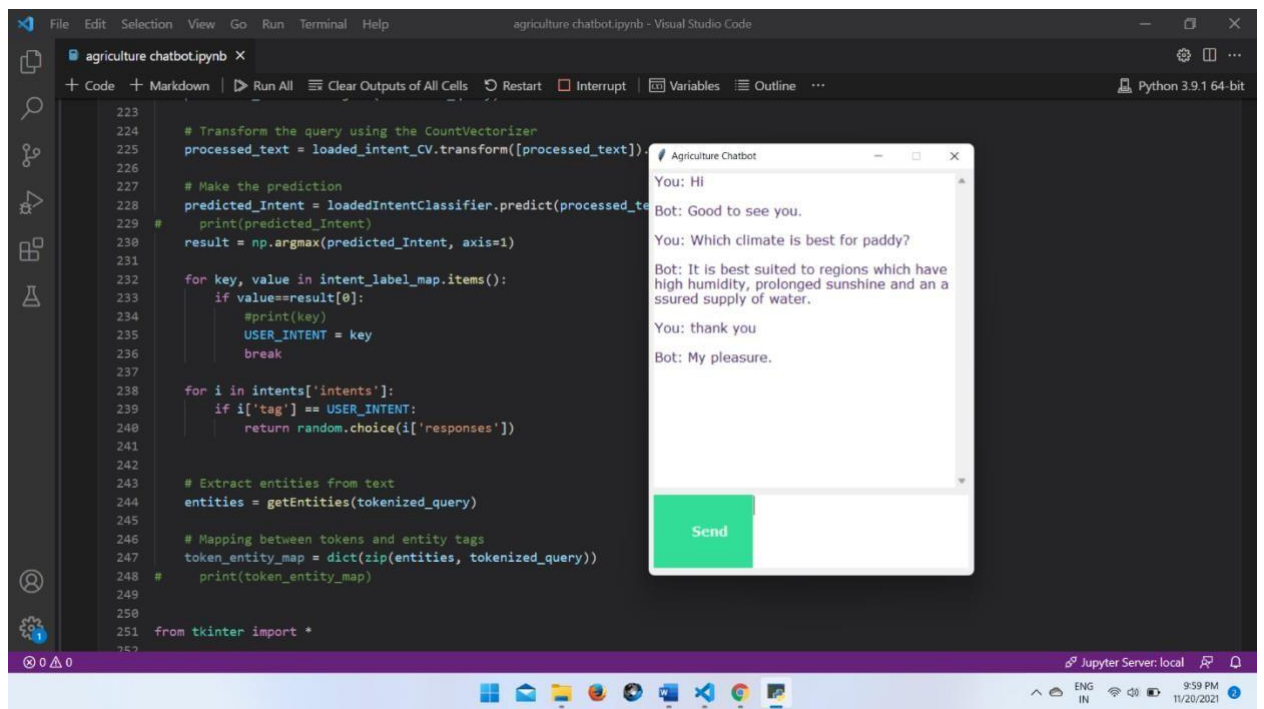
```

249
250 from tkinter import *
251
252 def send():
253     msg = EntryBox.get("1.0", 'end-1c').strip()
254     EntryBox.delete("0.0", END)
255
256     if msg != '':
257         ChatLog.config(state=NORMAL)
258         ChatLog.insert(END, "You: " + msg + '\n\n')
259         ChatLog.config(foreground="#442265", font=("Verdana", 12 ))
260         res = chatbot_response(msg)
261         ChatLog.insert(END, "Bot: " + res + '\n\n')
262
263         ChatLog.config(state=DISABLED)
264         ChatLog.yview(END)
265
266 base = Tk()
267 base.title("Agriculture Chatbot")
268 base.geometry("400x500")
269 base.resizable(width=FALSE, height=FALSE)
270
271 #Create Chat window
272 ChatLog = Text(base, bd=0, bg="white", height="8", width="50", font="Arial",)
273 ChatLog.config(state=DISABLED)
274 ChatLog.place(x=6,y=6, height=386, width=370)
275
276 #Bind scrollbar to Chat window
277 scrollbar = Scrollbar(base, command=ChatLog.yview, cursor="heart")
278 ChatLog['yscrollcommand'] = scrollbar.set
279 scrollbar.place(x=376,y=6, height=386)
280
281 #Create Button to send message
282 SendButton = Button(base, font=("Verdana",12,'bold'), text="Send", width="12", height=5,bd=0, bg="#32de97", activebackground=
283 SendButton.place(x=6, y=401, height=90)
284
285 #Create the box to enter message
286 EntryBox = Text(base, bd=0, bg="white",width="29", height="5", font="Arial")
287 EntryBox.place(x=128, y=401, height=90, width=265)
288
289 base.mainloop()

```

5.Output:





Reference:

- [1] FarmChat: A Conversational Agent to Answer Farmer Queries MOHIT JAIN, IBM * Research, India and University of Washington, Seattle, WA, USA PRATYUSH KUMAR, Indian Institute of Technology, Madras, India ISHITA BHANSALIQ. VERA LIAO, IBM Research, Yorktown Heights, NY, USA KHAI TRUONG, University of Toronto, Toronto, ON, Canada SHWETAK PATEL, University of Washington, Seattle, WA, USA
- [2] AgronomoBot: a smart answering Chatbot applied to agricultural sensor networks by GUSTAVO MARQUES MOSTAÇO 1, ÍCARO RAMIRES COSTA DE SOUZA 2, LEONARDO BARRETO CAMPOS 2, CARLOS EDUARDO CUGNASCA 1
- [3] AGRICULTURE ADVANCEMENT USING ARTIFICIAL INTELLIGENCE Kunal Vermal, Dinesh Pabbi², Avnish Singh Ja
- [4] <https://www.ijitee.org/wp-content/uploads/papers/v9i2S/B10081292S19.pdf>
- [5] <https://technotizerz.blogspot.com/2021/06/nlp-and-ann-based-college-enquiry-chat.html>
- [6] <https://www.ijrte.org/wp-content/uploads/papers/v8i2S5/B10370682S519.pdf>