

ONLINE FOOD ORDERS THROUGH WHATSAPP AUTOMATION BOT

Rimmalapudi Rajesh

Student, Department of Computer Science and Engineering(AI &ML), Vasireddy Venkatadri Institute of Technology, Guntur-522508, Andhra Pradesh, India.

Abstract – Due to rapid technological growth, automation bots are widely used to increase product sales. Most people are using food delivery apps like Swiggy, Zomato, etc., but those apps have some drawbacks. To overcome the drawbacks. I have developed a new solution for food delivery through WhatsApp Automation Bot. This WhatsApp Automation Bot is simply used to order food from famous Restaurants in a city. Mainly it is beneficial to unlettered people because we know that it is not easy to order food from Zomato or Swiggy etc for them. Nowadays most people are familiar with WhatsApp. So, they can easily place orders through WhatsApp.

Key Words: Automation, unlettered, Swiggy, Zomato, Bot.

1 INTRODUCTION

- What is a WhatsApp bot?

A WhatsApp chatbot is a computer program designed to automatically answer customer questions about your products and services, share content, and send notifications regarding orders, payments, and shipping on WhatsApp [5]. But with this WhatsApp automation bot, we can place an order through WhatsApp. Nowadays many members are familiar with WhatsApp [1]. So, they can easily place orders through WhatsApp [6]. I used Twilio for message handling API and MongoDB for database management.

I developed a WhatsApp Automation Bot using python and Twilio apart from sending messages will also use MongoDB to store messages in a database. For this, we created a relatively advanced multistep WhatsApp chatbot [2]. In this project all the services that we use are hosting, database, and messaging services all of these are completely free for certain limited conditions.

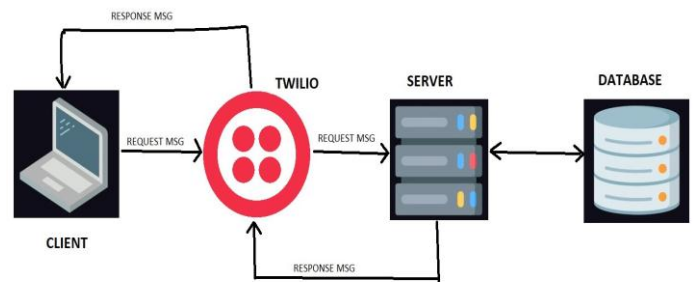
- Why WhatsApp?

WhatsApp is the most popular messaging tool and it has around 1.6 billion+ users, also the third most popular social media platform with 2 billion+ users. It's operational in over 180 countries and 60 languages. It's secure, fast, simple, and intuitive [8]. It helps people connect with friends and families across the globe, anytime. It's personalized and more intimate than other social media networks. It protects

the privacy of consumers thus, facilitating trust. It's free for users and doesn't need a technical setup. There are very few messaging applications like WhatsApp, and newer platforms aren't as effective or far-reaching.

2. SOFTWARE AND ARCHITECTURE

In this WhatsApp Automation Bot, I used web software. The software consists of carefully-organized instructions and codes written by programmers for different computer languages. Now, let us see in a diagrammatic way how this WhatsApp chatbot works.



Request-Response Cycle

In this, a client sends a request to the server through Twilio and the server sends the response to the Client through Twilio and the database acts as a stored and retriever of the data.

2.1 TWILIO

Twilio is a messaging API for SMS, MMS, and OTT messages operating a global network. It uses intelligent sending features to ensure messages reach end users wherever they are Twilio has SMS-enabled phone numbers available in more than 180 countries. For the free account, Twilio gives some number or if you want the specific number to pay for it [7]. The WhatsApp messages were sent to that number. Twilio is a single platform with flexible APIs for any channel, built-in intelligence, and global infrastructure to support you at scale. Twilio uses the power of the cloud to connect you with your supporters, beneficiaries, and volunteers, anywhere in the world [3]. This is the link below for how setup the Twilio: <https://github.com/RimmalapudiRajesh/Food-Delivery-through-whatsapp.git>

2.2 MONGODB

MongoDB is an open-source NoSQL database management program. NoSQL is used as an alternative to traditional relational databases. NoSQL databases are quite useful for working with large sets of distributed data. MongoDB is a tool that can manage document-oriented information, and store or retrieve information. It is a document database with the scalability and flexibility you want with the necessary querying and indexing. It enables you to store unstructured data and provides full indexing support and replication with rich and intuitive APIs [4].

2.2.1 CONFIGURING THE MONGODB

In MongoDB, I created a database named Restaurant with a collection of orders and users in it. In the "orders collection", the data that is required by the Restaurant owner like phone numbers, items, bills... etc is present, and in the "users collection", the messages that are sent by customers through WhatsApp are present.

This is the link below how to configure the MongoDB:

<https://github.com/RimlapudiRajesh/Food-Delivery-through-whatsapp.git>

3 INSTALL THE REQUIREMENTS

Install python from google. Also, install the pycharm community version from google. Install the NPM (Node Package Manager). This comes bundled with node.js (node.js is used in javascript runtime for the general to develop the backend applications) and after the installation process check in the command prompt whether the node.js is installed or not. Now we also install the npm packages like nodemon (which helps us to restart the server in case we do any changes to it) and localtunnel (It makes your server accessible online). For this type of command prompt as npm install -g nodemon localtunnel where -g determines to install the packages globally. Also, install the python package like a flask, twilio, pymongo, and dnspython. For this type of command prompt as pip install flask twilio pymongo dnspython

3.1 SOFTWARE TOOLS

The requirements to develop the project are:

flask: This is used to create a web server

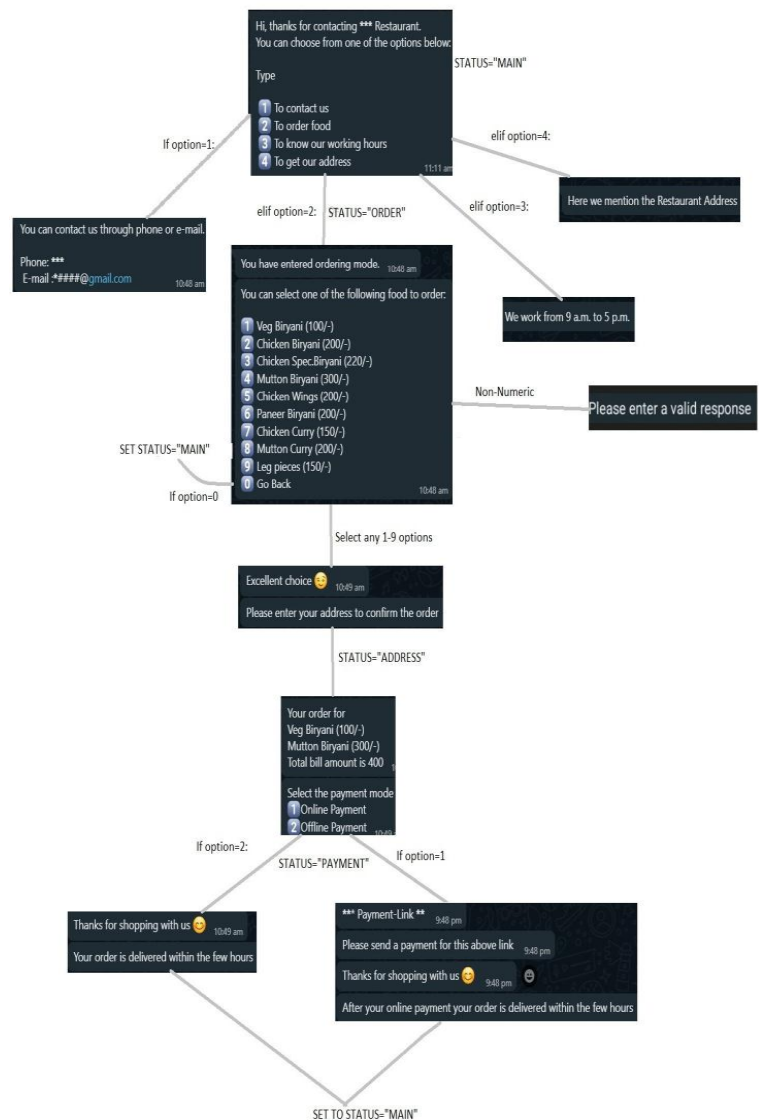
twilio: Which is a messaging API (Application Programming Interface)

pymongo: This is used to interact with the database

dnspython: which is Twilio dependency on this particular package.

4 FLOW CHART

This is the flowchart of how the steps are constructed to implement the WhatsApp Automation Bot for Food Delivery.



5 SOURCE CODE

This is the source code to build a WhatsApp Automation Bot for food delivery.

<https://github.com/RimlapudiRajesh/automate-whatsapp>

```

1 from flask import Flask, request
2 from twilio.twiml.messaging_response import MessagingResponse
3 from pymongo import MongoClient
4 from datetime import datetime
5 cluster = MongoClient("mongodb+srv://raghu:raghu@cluster0.guea2.mongodb.net/myFirstDatabase?retryWrites=true&majority")
6
7 db = cluster["Restaurant"]
8 users = db["users"]
9 orders = db["orders"]
10
11 app = Flask(__name__)
12 @app.route("/", methods=["get", "post"])

```

```

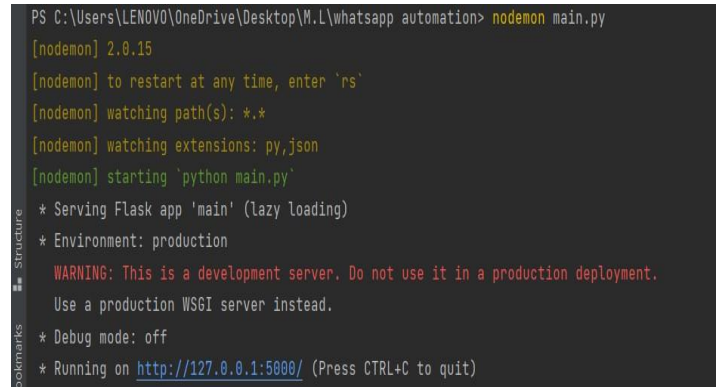
11 app = Flask(__name__)
12 @app.route("/", methods=['get', 'post'])
13 def reply():
14     bill = 0
15     text = request.form.get("Body")
16     number = request.form.get("From")
17     number = number.replace("whatsapp:", "")
18     res = MessagingResponse()
19     user = users.find_one({"number": number})
20     if bool(user) == False:
21         msg = res.message("Hi, thanks for contacting ***** Restaurant.\nYou can choose from one of the options below: "
22             "\n\nType\n 1 To contact us\n 2 To order food\n 3 To know our working hours\n 4 "
23             "To get our address")
24         users.insert_one({"number": number, "status": "main", "messages": [], "bill": 0})
25     elif user["status"] == "main":
26         try:
27             option = int(text)
28         except:
29             res.message("Please enter a valid response")
30             return str(res)
31
32     if option == 1:
33         res.message("You can contact us through phone or e-mail.\n\nPhone: ***** \n E-mail: *****@gmail.com")
34     elif option == 2:
35         res.message("You have entered ordering mode.")
36         users.update_one({"number": number}, {"$set": {"status": "ordering"}})
37         res.message("You can select one of the following food to order: \n\n 1 Veg Biryani (100/-)
38             "\n 2 Chicken Biryani (200/-) \n 3 Chicken Spec.Biryani (220/-) "
39             "\n 4 Mutton Biryani (300/-) \n 5 Chicken Wings (200/-) \n 6 Paneer Biryani (200/-)
40             "\n 7 Chicken Curry (150/-) \n 8 Mutton Curry (200/-) \n 9 Leg pieces (150/-) \n 0 Go Back")
41     elif option == 3:
42         res.message("We work from 9 a.m. to 5 p.m.")
43
44     elif option == 4:
45         res.message("Here we mention the Restaurant Address")
46     else:
47         res.message("Please enter a valid response")
48     elif user["status"] == "ordering":
49         try:
50             option = str(text)
51         except:
52             res.message("Please enter a valid response")
53             return str(res)
54         if option == '0':
55             users.update_one({"number": number}, {"$set": {"status": "main"}})
56             res.message("You can choose from one of the options below: "
57                 "\n\nType\n 1 To contact us\n 2 To order food\n 3 To know our working hours\n 4 "
58                 "To get our address")
59         elif option=="Hi":
60             x=""
61             bill=0
62             for i in option:
63                 if '1' <= i <= '9':
64                     Food = ["Veg Biryani (100/-)", "Chicken Biryani (200/-)", "Chicken Spec.Biryani (220/-)",
65                         "Mutton Biryani (300/-)", "Chicken Wings (200/-)", "Paneer Biryani (200/-)",
66                         "Chicken Curry (150/-)", "Mutton Curry (250/-)", "Leg piece (150/-)"]
67                     selected = food[int(i) - 1]
68                     x+=selected+"\n"
69                     bill+=int(selected[-6:-3])
70             users.update_one({"number": number}, {"$set": {"status": "address"}})
71             users.update_one({"number": number}, {"$set": {"item": x}})
72             users.update_one({"number": number}, {"$set": {"bill": bill}})
73             res.message("Excellent choice 🍽️")
74             res.message("Please enter your address to confirm the order")
75         else:
76             res.message("Please enter a valid response")
77
78     elif user["status"] == "address":
79         x = user["item"]
80         bill=user["bill"]
81         res.message(f"Your order for \n{x}Total bill amount is {bill}\n")
82         orders.insert_one({"number": number, "item": x, "bill": bill, "address": text, "order_time": datetime.now()})
83         users.update_one({"number": number}, {"$set": {"status": "payment"}})
84         res.message("Select the payment mode\n 1 Online Payment\n 2 Offline Payment\n")
85     elif user["status"]=="payment":
86         if text == "2":
87
88             res.message("Thanks for shopping with us 🍽️")
89             res.message("Your order is delivered within the few hours")
90         elif text == "1":
91
92             res.message("***** Payment-Link *****")
93             res.message("Please send a payment for this above link")
94             res.message("Thanks for shopping with us 🍽️")
95             res.message("After your online payment your order is delivered within the few hours")
96         else:
97             res.message("Please enter a valid response")
98             users.update_one({"number": number}, {"$set": {"status": "ordered"}})
99     elif user["status"] == "ordered":
100         res.message("Hi, thanks for contacting again.\nYou can choose from one of the options below: "
101             "\n\nType\n 1 To contact us\n 2 To order food\n 3 To know our working hours\n 4 "
102             "To get our address")
103
104         users.update_one({"number": number}, {"$set": {"status": "main"}})
105         users.update_one({"number": number}, {"$push": {"messages": {"text": text, "date": datetime.now()}}})
106         return str(res)
107
108
109 if __name__ == "__main__":
110     app.run()

```

5.1 EXECUTION OF THE SOURCE CODE IN PYCHARM

To execute the source code in Pycharm follow these steps in the terminal:

(1) Firstly, divide the terminal into two parts. In the first part type “nodemon filename” and then press enter.



(2) In the second part type as nodemon --watch “filename” --exec “It –subdomain YOURSUBDOMAIN –port 5000” –delay 2 here,

nodemon: used to restart the server when we do any changes to it.

--watch: used to watch the changes if we do any changes in the source code.

filename: Enter a filename.

--exec: used to execute the changes we made in the source code.

It (local tunnel): Whenever we create a server in the system it stays local to your IP address. The outsiders of the people can’t access the server. what the local tunnel does is it assigns you a website URL and when the website URL is queried. It proxies all the requests to the local server. In other words, it kind of makes the servers accessible online.

--subdomain: used for sub-domain.

YOURSUBDOMAIN: enter the unique subdomain here like mango-2006-apple...etc which gets the specific URL. Make sure that it must be unique.

--delay 2: which helps us to the gap between the server restarting. If we don’t give any gap between the server restarting the local tunnel assume that the subdomain is already taken and will assign some random subdomain which is not what we want. In this, I gave 2 seconds gap between the server restarting.


```
PS C:\Users\LENOVO\OneDrive\Desktop\M.L\whatsapp automation> nodemon --watch "main.py" --exec "lt --subdomain unique-name-2006 --port 5000" --delay 2
[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): main.py
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `lt --subdomain unique-name-2006 --port 5000`
your url is: https://unique-name-2006.loca.lt
[nodemon] clean exit - waiting for changes before restart
```

(3) This URL is pasted on the Twilio sandbox. The messages are exchanged through this URL. This URL must be unique for that we mention a unique subdomain name.

6 RESULT AND DISCUSSION

In WhatsApp the chat between the customer and WhatsApp bot as in fig 6.1.1:

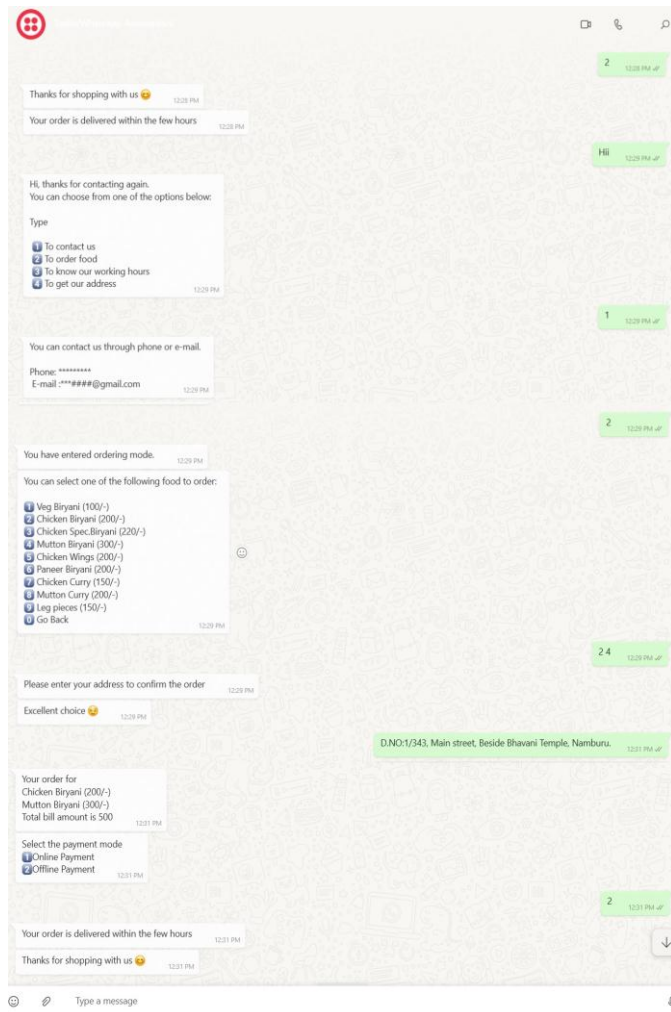


Fig 6.1.1: WhatsApp Bot conversion with customer

The information that is needed by the Restaurant owner, like phone number, items, bills... etc. as shown in fig .6.1.2, are displayed on the “orders collection” of MongoDB, and the remaining information like options selected by the users, is stored in the “users collection”.

This is how the restaurant owner sees the only needed information to deliver the food to that address.

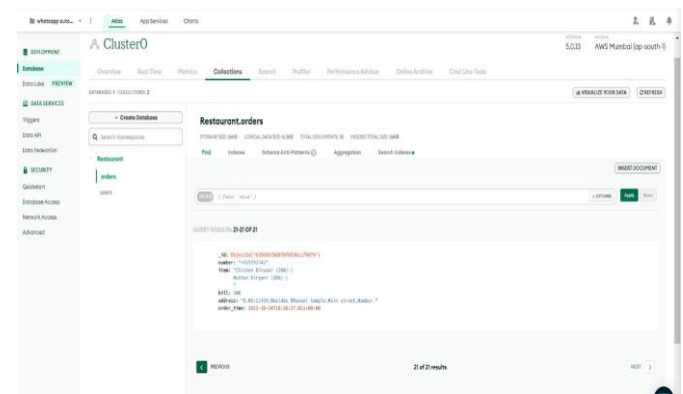


Fig 6.1.2: orders collection in MongoDB

7 CONCLUSION

After the successfully executed then deploy this project in Heroku. This paper presents how to make a food order through WhatsApp Automation Bot. I successfully developed and implemented the software to order food from Restaurants. This model was developed and implemented at a low cost and very cheap offering to all kinds of people because it stores the data in a database. In the future, Bot has good development to increase sales production in any sector. Maintenance cost still decrease by hosting this application’s data on the cloud.

REFERENCES

[1] Roberta Roberta De Cicco, Susana Cristina Lima da Costa e Silva & Francesca Romana Alparone (2021) “It’s on its way”: Chatbots applied for online food delivery services, social or task-oriented interaction style, Journal of Foodservice Business Research, 24:2, 140-164
DOI: [10.1080/15378020.2020.1826268](https://doi.org/10.1080/15378020.2020.1826268)

[2] Rajendiran Anbumathi, Sriram Dorai and Umaya Palaniappan, Evaluating the role of technology and non-technology factor in influencing brand love in Online Food Delivery services.vol 71,2023,103181.

[3] Sujana A S, D R Ramesh Babu, S Venkatesan, Twilio Integration with Dialogflow for Effective Communication. Journal of Web Development and Web Designing powered by MAT journal, vol 4, issue 2, 2019
<https://core.ac.uk/download/pdf/230494953.pdf>

[4] Kyle Banker, Douglas Garrett, Petter Bakkum, Tim Hawkins and Shaun Verch wrote a book about the complete details of MongoDB Version 3.0 published by manning publications, 2016

[5] Darius Zumstein and Sophie Hundertmark, Chatbots-Technology for Personalized Communication, Transactions, and services on IADIS International Journal on WWW/Internet.2017, vol 15, 96-109.

[6] Verloop.io, WhatsApp for Restaurants 101: WhatsApp Food Ordering System accessed on Oct 2022.
<https://verloop.io/blog/whatsapp-business-restaurants/>

[7] Twilio Programmable API for WhatsApp
<https://www.twilio.com/docs/whatsapp>

[8] <https://www.digitaltrends.com/mobile/what-is-whatsapp/>

BIOGRAPHY



¹ Rimmlapudi Rajesh

Studying in BTech 3rd year, Department of Computer Science and Engineering (AI& ML), Vasireddy Venkatadri Institute of Technology, My interests are Artificial Intelligent and Machine Learning.