ROBODOLPH

A holiday Odd Jobs AI Chatbot

Serveless Guru 2023 Hackathon

By Muaaz Bayat

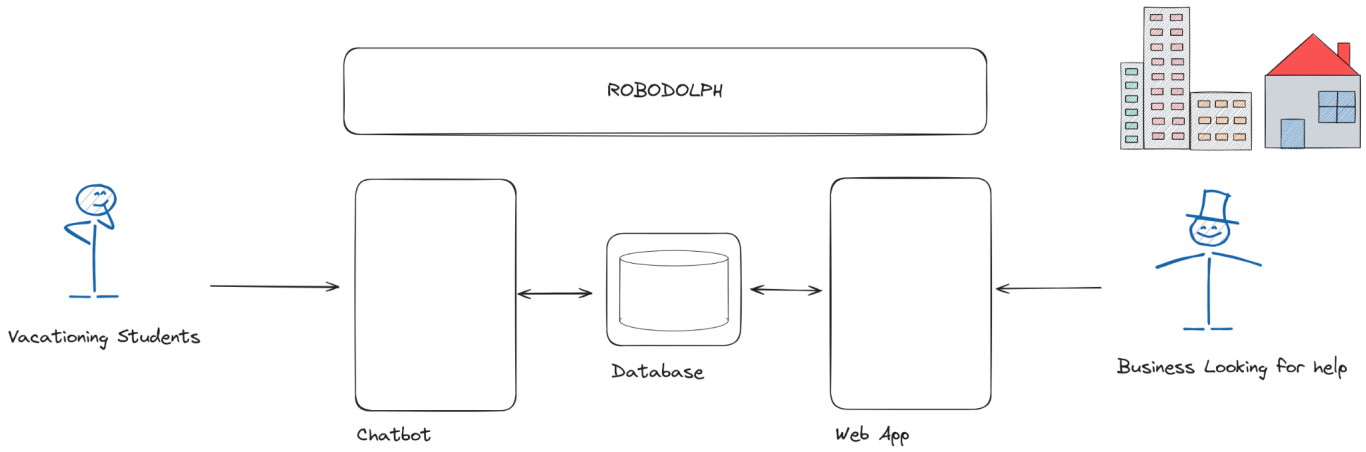
00: THE IDEA

The task issued in the onset of the hackathon was to build a holiday themed chatbot that incorporates some sort of large languge model.

The idea that I have selected after an initial brainstorming process is a GPT-3 powered chatbot called robodolph. Robodolph helps vacationing students to view tasks created by busy households and businesses.

During the holiday, schools are out and businesses and households are generally busier than throughout the year. This means that there is a supply and demand of labour. Supply is driven by the amount of students looking for work. And demand is driven by the business and households requiring assistance in this peak period.

Robodolph is aimed to match this supply and demand. It will allow businesses and households to create tasks and holiday job openings. This will be done on a web app. Robodolphs chatbot will also allow vacationing students to view tasks and jobs in their area based on their location. Engagement between the two parties will not happen on the platform.



01: THE CHALLENGE

The organisers have announced the following criteria on how the project will be judged.

Judging Criteria					
Originality	0.3	Innovation	0.35	Implementation	0.35

Extrapolating upon the criteria, we can lay out the following stakeholder requirements.

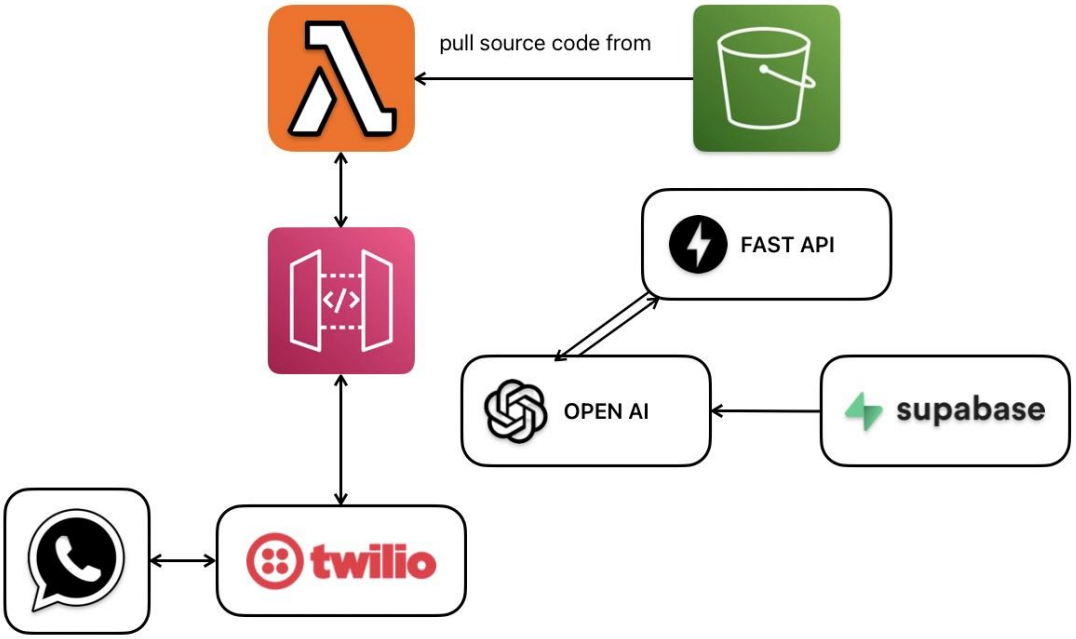
Project Requirements		
Code	**Requirement**	**Detail**
PR:01	Back End Architecture	
01.1	Structurally Sound	Thorough Testing and Edge cases handled well.
01.2	Scallable	System designed to handle scale (100k users)
01.3	Efficient	no process should run > O(n^2)
PR:02	Article Completeness	
02.1	Thoroughness	Complete in Detail
02.2	Clarity	Intentional in content
02.3	Comprehensiveness	Concise in diction
PR:03	Real World Applicability	
03.1	Reach	How many will this affect
03.2	Delta of innovation	How much will this change
PR:04	Code Quality	
04.1	Structured Proccess	Code must be planned
04.2	Best Practices Adhered	nomenclature and testing

From these stakeholder requirements and the idea of the project, we can lay out the following functional requirements. The combination of project and stakeholder requirements will determine the success of the project.

Functional Requirements		
Code	**Requirement**	**Detail**
FS1	Students should be able to:	
1.1	view tasks	
1.2	sort tasks by location	
1.3	engage with bot on tasks	
FHB2	Households and business be able to:	
2.1	sign up	
2.2	sign in	
2.3	create tasks	

02: THE ARCHITECTURE

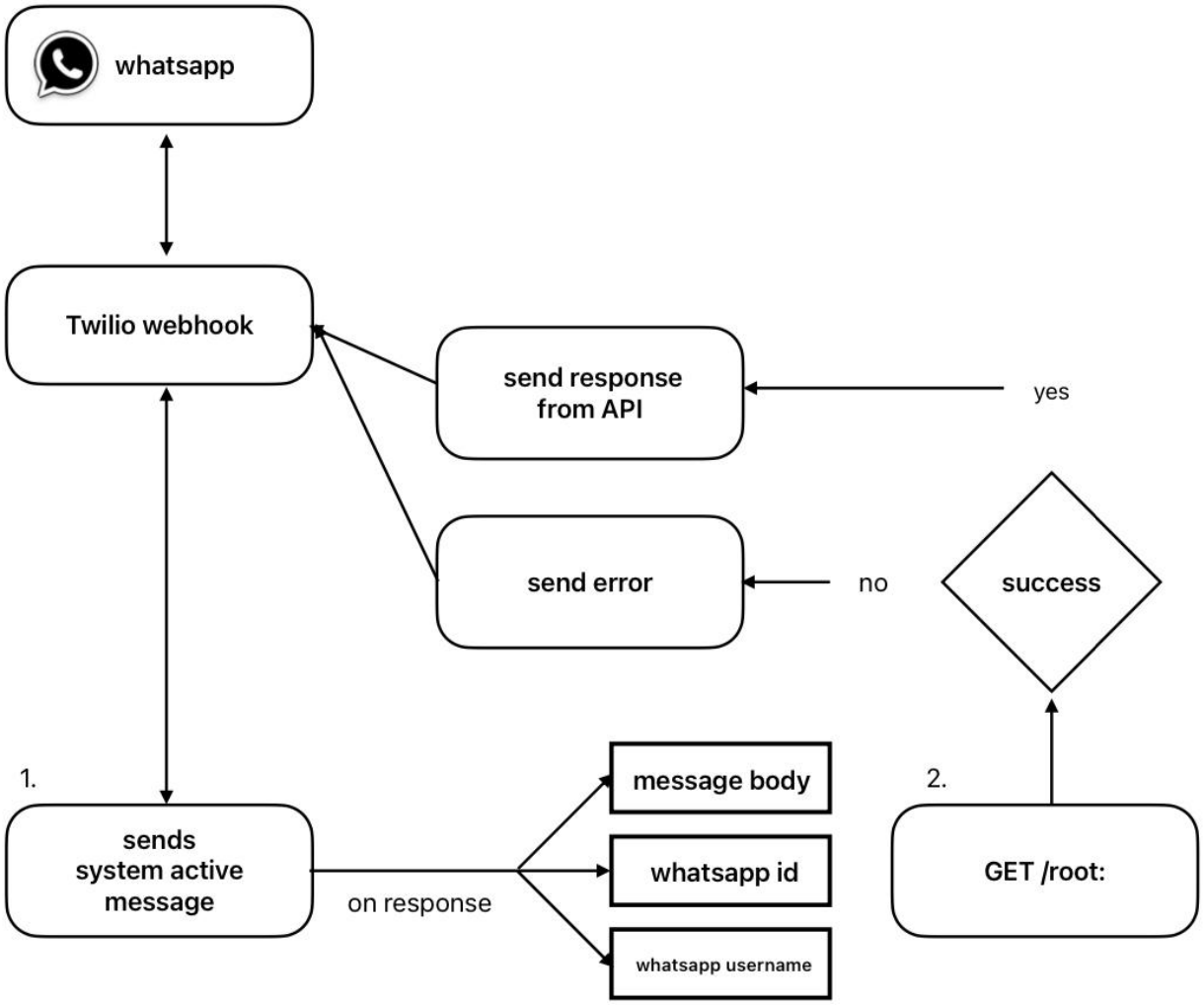
Vacationing students can interact with robodolph via whatsapp. I am using Twilio's sanbox to connect to whatsapp. My custom API is built with FastAPI and is deployed on a AWS Lambda function via a S3 bucket holding the source in a zip. The API connects to Open AI's assistants and also a Supabase Postgres DB for data persistence.



The diagram above shows how the system works from a high-level.

03: KEY ALGORITHMS

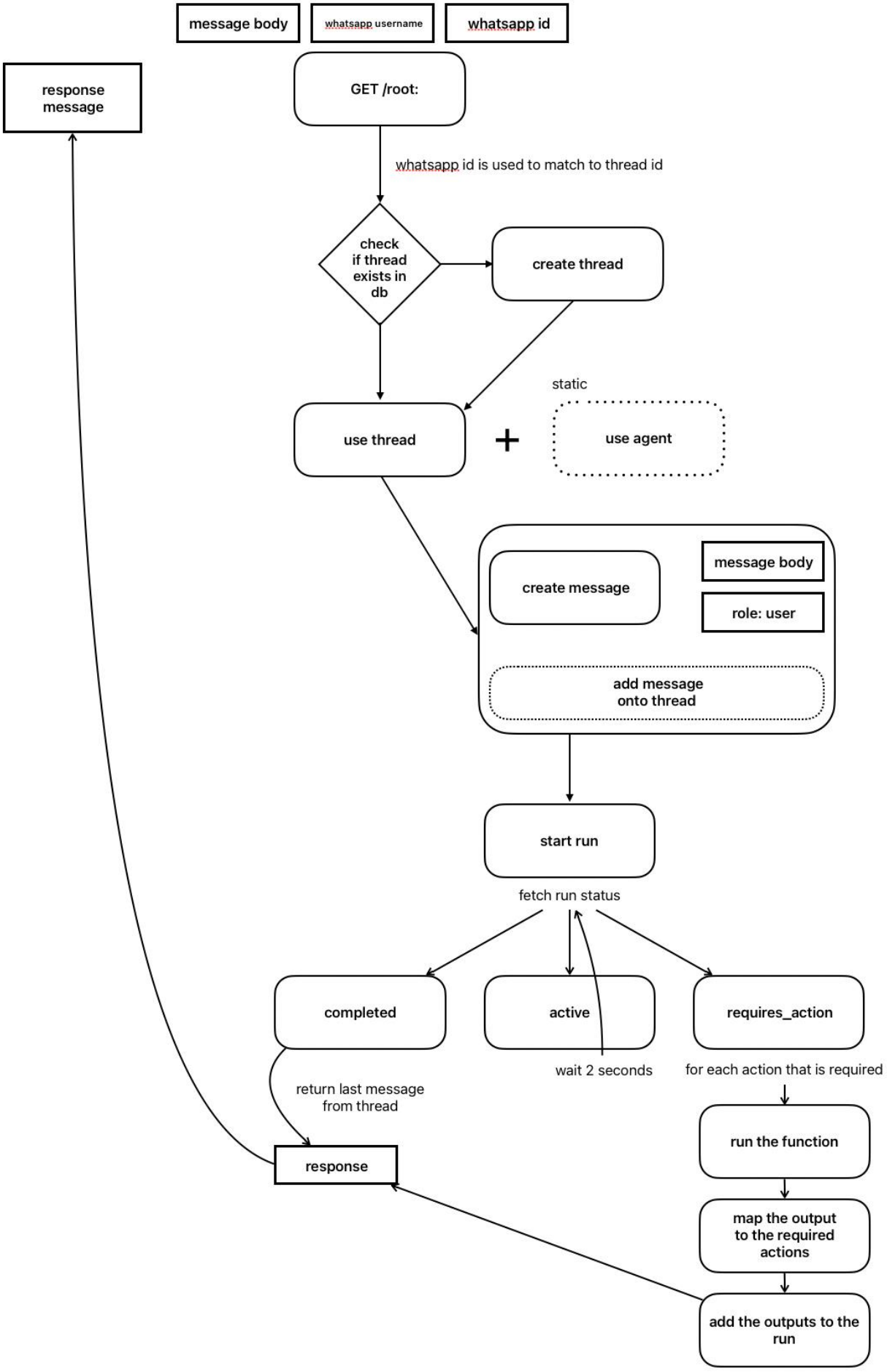
Cyclical API response flow



When a message is received, whatsapp sends a webhook event to twilio. Twilio then responds with a message to the user saying that the system is active. When a user responds to this message, parameters are extracted and a call is made to our custom API on the root endpoint.
If the response is a success, the message from the API is sent to the user, else the error message is sent.

Note: we are aware that the system active message takes away from the UX. It is a necessary step as Twilio can only extract parameters on response :(
With time we would have implemented a check to only send the active message on the first engagement with the user. To do it as an introductory message and to extract parameters from persistent context.

API Root Endpoint



This flowchart outlines the process for generating a response with the help of the LLM.
Here's a breakdown of the steps:

1. A user sends a message which includes the message body, their WhatsApp username, and WhatsApp ID.
2. A GET request is made to the /root endpoint, with the WhatsApp ID provided.
3. The system then checks if this WhatsApp ID matches an existing thread ID in the database.
4. If there is no existing thread, a new one is created.
5. If a thread exists (or once the new thread is created), that thread is used for further operations.
6. A message object is created with the message body and the role of the sender set as 'user'.
7. This message is then added onto the existing or new thread.
8. With the thread updated, the system starts a 'run', which is a process which open ai uses to generate responses.
9. The system periodically fetches the status of the run.
10. If the run is 'active', the system waits for 2 seconds before checking the status again.
11. If the run requires action, each necessary action is executed.
12. For each action, a function is run, and the output of this function is mapped to the required actions.
13. These outputs are then added to the run, as an update to the thread.
14. Once the run is 'completed', the last message from the thread is returned as the response.
15. Finally, this response is sent back to the user, completing the interaction cycle.

04: THE RESULTS

05: DEMOS

To chat to **robodolph live**:

1. Scan the QR code, and send the “join chair-day” message
2. Say “hi” to robodolph, He should say that he is there and active
3. Ask him to tell you about himself, say something like “what is it that you do?”
4. Ask him to help you find something to do in x city this holiday.

Twilio WhatsApp Sandbox

OR

+14 155238886

send message “join chair-day”

Click this link to use whatsapp web

To see an **pre recorded demo** of **both the chat and the webapp** you can watch this video here:

Demo Video Link

Twilio WhatsApp Sandbox

OR

+14 155238886

send message “join chair-day”

Click this link to use the webapp

06: ACKNOWLEDGEMENTS

Big shout-out to everyone on this learning adventure with me – it's been a wild ride, and I've grown a ton. Hats off to the folks at Serverless Guru for the opportunity, cool tricks and hacks. And to the crew pulling the strings together – you guys rock! Here's to more code, less bugs, and great vibes ahead!

Article written by: Muaaz Bayat with some help from Chat GPT