How do we run a web scraping code every time we update the website?

**Ans:** We can create triggers on the database itself so that whenever changes are made that trigger is caught by the system or some middle agent which leads to execution of the webscraping code. Thus, leading to a set trigger task that leads to running of webscraping code.

Steps:

1. Create a database trigger that tracks the changes in the database table that you want to scrape.
2. Create a webhook that will be notified whenever the trigger is executed.
3. In the webhook, use a web scraping library to extract the data from the website.
4. Save the webhook.

A webhook is a way for one system to send a notification to another system when an event occurs. The notification is sent as an HTTP request, and the payload of the request contains information about the event that occurred.

Webhooks are a powerful way to connect two systems and automate tasks. They are often used to send notifications, update data, or trigger other actions.

Benefits of using webhooks:

* Real-time notifications: Webhooks can be used to send real-time notifications, so that you are always up-to-date on the latest events.
* Automated tasks: Webhooks can be used to automate tasks, so that you don't have to manually keep track of events.
* Decoupled systems: Webhooks can be used to decouple systems, so that they can communicate with each other without having to be tightly coupled.

How do we deal with constant updation of the source file how does the chatbot responds to it?

**Ans:** Commonly there are 3 methods that can be used to deal the above situation.

1. Retraining the chatbot (most favourable in our case): We can retrain the chatbot on the updated data in the source file.
   1. Pros:
      1. Ensures that the chatbot has the latest information.
      2. Can be used to add a new feature to the chatbot.
   2. Cons.
      1. Time-consuming (too much if done regularly)
      2. Expensive
      3. Can be disruptive to the chatbot’s users.
2. Using a Webhook: The chatbot can use the notifications sent by the webhook to update its knowledge base.
   1. Pros:
      1. Easy to set up
      2. Can be used to update the chatbot without disrupting its users.
   2. Cons:
      1. My not be able to keep the chatbot up-to-date with the latest information.
3. Searching the source file: chatbot can also search for information from the source file. When a user asks a question the chatbot can search the source file for that question.
   1. Pros:
      1. Simple to implement.
      2. Can be used to update the chatbot without disrupting its users.
   2. Cons:
      1. May not be able to find the answer to the user’s questions
      2. May not be able to keep the chatbot up-to-date with the latest information.

Ideas:

* We could structure the data into different source files for different intent and segregate the data according to the intent, this would help in case the data is updated for a specific data we don’t have to change or possible retrain the chatbot again on information that have not been changed. This will help in less time consumption in retraining the whole chatbot on all the whole source file if possible. More characterised the data is more beneficial for the chatbot in terms of Time complexity and the expensive operation of retraining the chatbot.

Problem with using Webhook to update the knowledge base of the chatbot?

Webhook is way for communication between 2 systems. If let’s say the database is changed in one system which triggers a webhook which sends a notification to other system in case the connection latency between the 2 systems is high it may lead to a longer delay and sometime discarding of the notification which will lead the chatbot to still show the discarded information.

The above problem can be minimized by simply ensuring that the connection between the 2 systems has low latency.

Another method is to use a webhook that supports long polling. Long polling is a technique that allows the chatbot to keep the connection to the system that is sending the notification open. This means that the chatbot will not miss any notifications, even if there is a delay in the network.

Here are some additional tips to help you keep your chatbot up to date:

* Use a webhook that supports long polling. This will ensure that the chatbot does not miss any notifications, even if there is a delay in the network.
* Set up a regular schedule to check the knowledge base for updates. This will help to ensure that the chatbot is always up to date with the latest information, even if there are no notifications.
* Use a content management system (CMS) to manage the knowledge base. This will make it easier to update the knowledge base and to track changes.
* Train the chatbot to recognize when the information in the knowledge base is outdated. This will help the chatbot to provide users with the most accurate information possible.

Conclusion:

Here we learnt different methods of updating the data source as well as how to integrate it with chatbot. This will help in reducing the computation as well as reducing the disruption for a user.