[Information 1](#_Toc324444557)

[Introduction 1](#_Toc1630430329)

[Telegram 1](#_Toc1233748484)

[What is a telegram bot? 1](#_Toc281767240)

[What problems can be solved by a Telegram bot? 2](#_Toc160978542)

[Additional files 2](#_Toc159986100)

[Database 2](#_Toc1019761263)

[Introduction to the theory of ORM-models 2](#_Toc1534101494)

[user.py 3](#_Toc1837422196)

[all\_models.py 4](#_Toc1580178789)

[db\_session.py 4](#_Toc1033340285)

[Main code 7](#_Toc1378147963)

[Functions 7](#_Toc1377267822)

[Gpt 7](#_Toc1359014316)

[Links 8](#_Toc1853744976)

[Quote 9](#_Toc1531378185)

[Links 10](#_Toc137486635)

[Wikipedia 10](#_Toc479220744)

[Links 11](#_Toc747373209)

[Weather 11](#_Toc179557292)

[Links 14](#_Toc772468241)

[News 14](#_Toc1770296459)

[Links 16](#_Toc1175289666)

[Recipes 16](#_Toc1766860471)

[Links 18](#_Toc1708657721)

[Events 19](#_Toc1373291740)

[Links 21](#_Toc571984347)

[Flight 21](#_Toc2103142183)

[Links 22](#_Toc1352226970)

[User’s profile 23](#_Toc502750845)

# Information

**GIT-HUB link -** [**https://github.com/ArtSN7/telegram-bot**](https://github.com/ArtSN7/telegram-bot)

**Documentation of how to write telegram-bots with theory and examples, you might find in the repository in the file GUIDE.md. It will be useful for you to check it first before reading documentation, as it will help you deeply understand how does it all work.**

# Introduction

## Telegram

Telegram is a messenger (messaging program) implemented using a client-server architecture. Using the server to create a dialogue between two clients, Telegram sends text messages through it or directly, as well as images, videos, or documents in other formats.

## What is a telegram bot?

A Telegram bot is a special user whose behaviour is controlled by some program. Technically, it makes no difference to the server whether a given user is a human or a bot: to the server, both clients look the same.

## What problems can be solved by a Telegram bot?

Everything, it is limited only by imagination. Here are some examples:

Autoresponders - all situations where an unambiguous answer to a request is required. For example, the bot can provide telephone numbers and other contacts of the organization, its working hours, or provide other background information upon request

Interface for accessing web services - the bot can make requests to various APIs *(application programming interface, which is a set of definitions and protocols for building and integrating application software)* andsend responses in the form of telegram messages.

\*More about API –

- <https://www.redhat.com/en/topics/api/what-are-application-programming-interfaces>

- <https://www.ibm.com/topics/api>

Action scenarios - the bot can go through any scenario, ask the user certain questions and collect answers to them. For example, when registering in any service or when applying for a service

Games - the bot can send pictures, so you can create any games that do not require an instant response, such as chess or different card games ( example - @anicardplaybot )

-------------------------------------------------------------------------------------------------------------------------------------

# Additional files

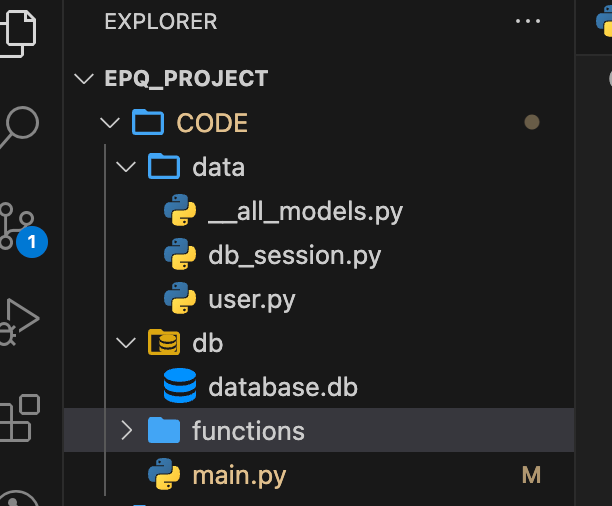
-------------------------------------------------------------------------------------------------------------------------------------

## Database

### Introduction to the theory of ORM-models

Imagine that you have an object at your disposal that is linked to a database. This object takes over all the work of organizing communication with data. All you must do is give it commands: get data, filter them according to a given condition, write data, etc., and converting commands into SQL queries is already the object's concern.

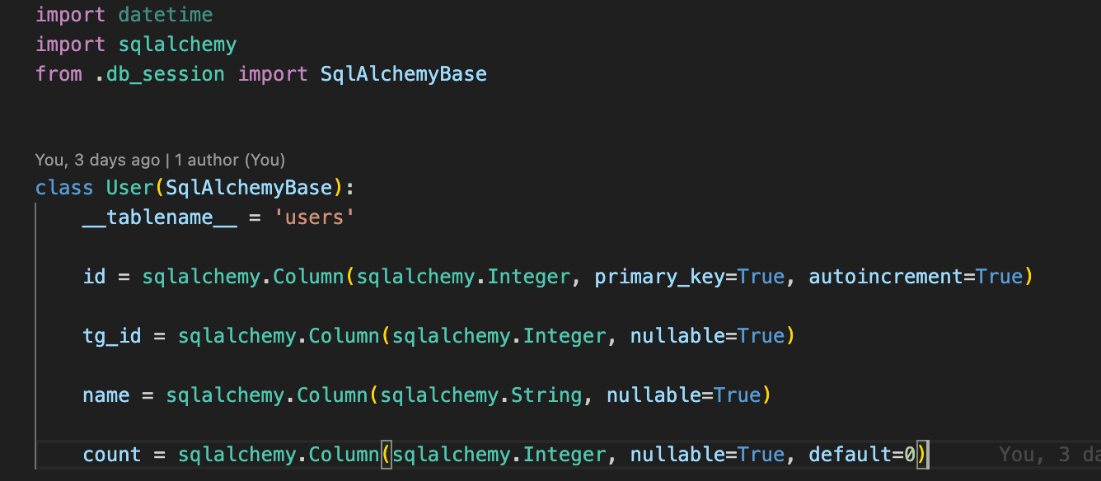
In large applications, ORM (Object-Relational Mapping) technology is often used — a layer that allows working with a database through language objects. In addition, most ORMs allow you to generate database migration scripts to maintain versioning (remotely comparable to git, but for databases), and provide the developer with a lot of other useful functionality. We will use the sqlalchemy library ( documentation - <https://www.sqlalchemy.org> ). It can be used not only when creating web applications, but also when developing any programs that interact with databases.



db — to store a single database file

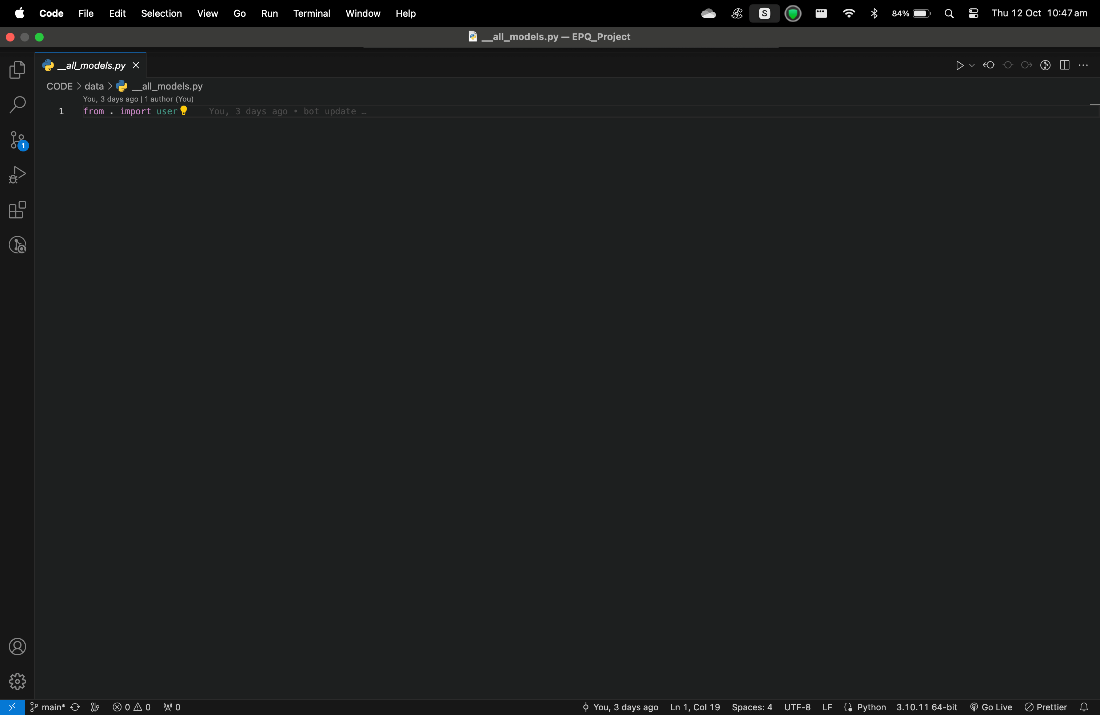
data — to store the classes needed to interact with the database

### user.py



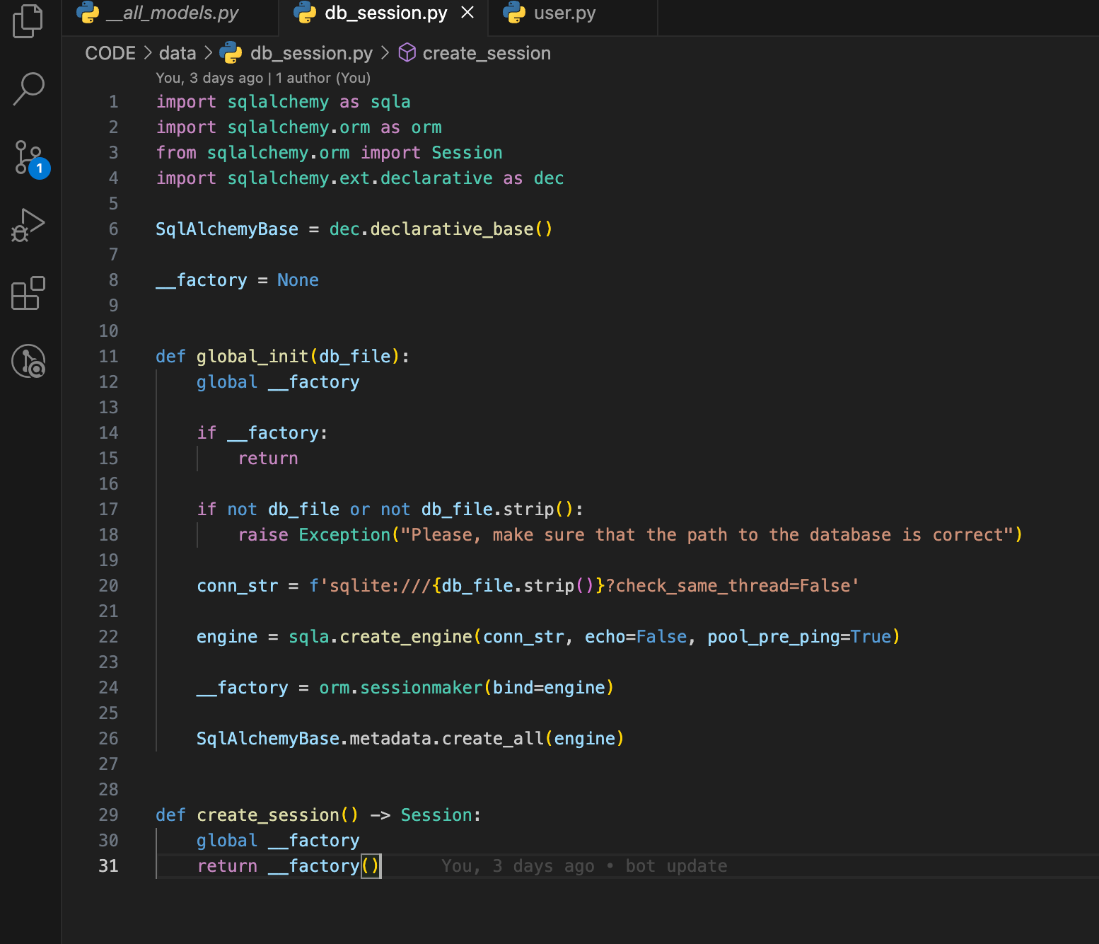
It describes User class in which there are columns with information from database. For example, field “name = sqlalchemy.Column(sqlalchemy.String, nullable=True) “means that there is a column with names which consists of String values.

### all\_models.py



we are creating a connection “factory” to my database that will work with the engine we need.

### db\_session.py



This file will be responsible for connecting to the database and creating a session to work with the database.

First, we import the necessary extensions — the sqlalchemy library itself, then the part of the library that is responsible for the ORM functionality, then the Session object responsible for connecting to the database, and the declarative module — it will help to declare database.

Let us create two variables: SqlAlchemyBase — some abstract declarative database into which we will later inherit all our models, and \_\_factory, which I will use to get connection sessions to our database.

Also, in the db\_session file.py I will need to make two more functions global\_init and create\_session.

global\_init takes the database address as input, then checks if I have already created a connection factory (that is, if I am not calling the function for the first time).

I check that I have been given a non-empty database address, and then create a conn\_str connection string (it consists of the database type, the address to the database and connection parameters), which I pass to Sqlalchemy so that it.

I chose the right database engine (engine variable). In my case, it will be an engine for working with SQLite databases.

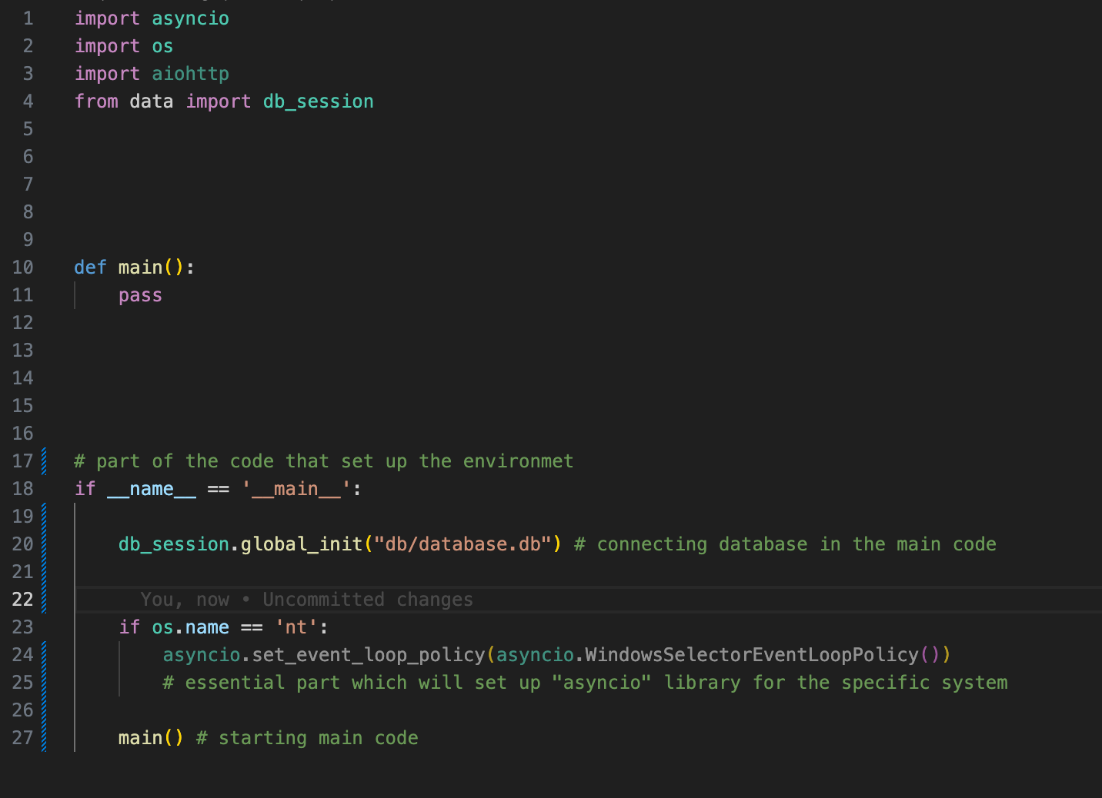
Moving further, the create\_session function is needed to get a connection session to our database.

Then you need to add import of the contents of the db\_session file to the main code:

from data import db\_session

And before launching the app.run() application, we will add a call to the global initialization of everything related to the database:

db\_session.global\_init("db/database.db")



-------------------------------------------------------------------------------------------------------------------------------------

# Main code

-------------------------------------------------------------------------------------------------------------------------------------

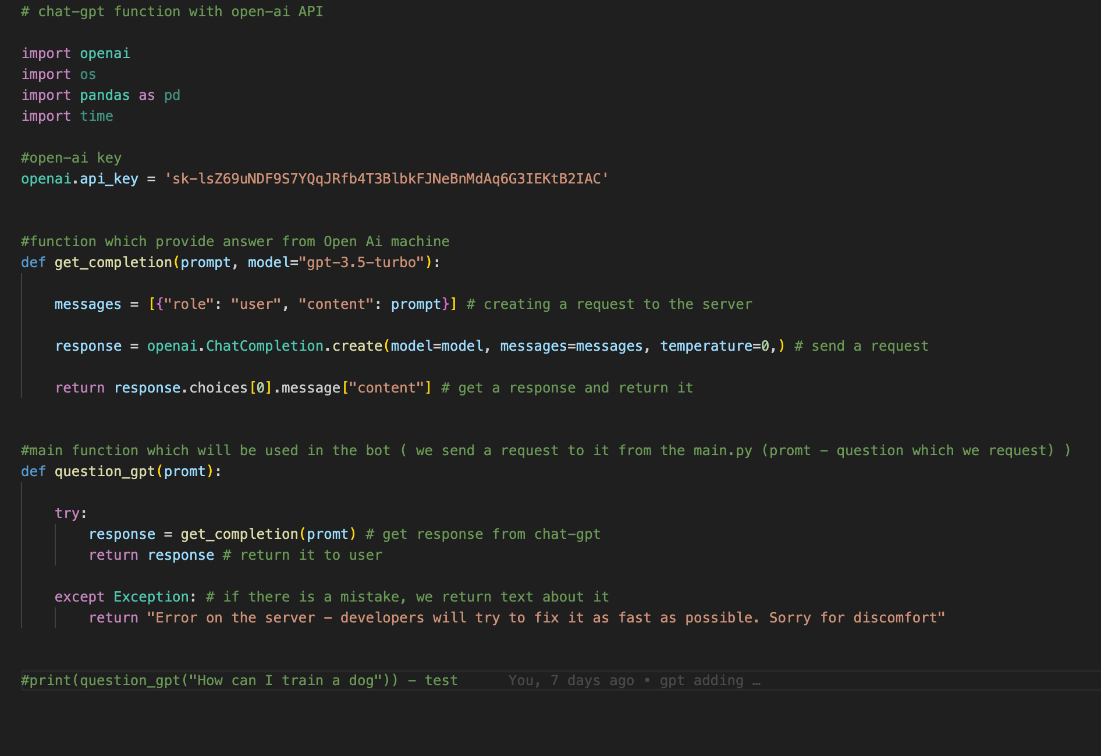
**// main code explanation**

-------------------------------------------------------------------------------------------------------------------------------------

# Functions

-------------------------------------------------------------------------------------------------------------------------------------

## Gpt

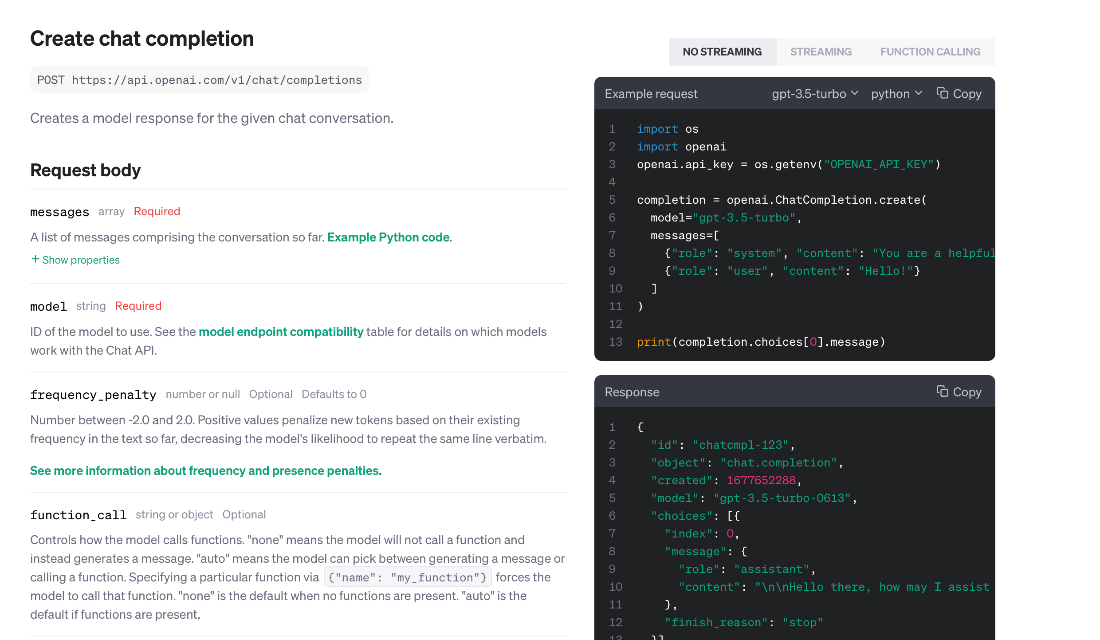


Gpt function provides to user an opportunity to ask questions to the Open-AI creation called “Chat-GPT” ( which is currently 3.5 ) . In this function, I use their API, which give me an access to their machine which can analyse data. Therefore, user send a request (question), it goes to question-gpt function, then the question is being sent to the server using get\_completion, and in the end being returned to the user.

“Question-gpt" connects message sent by user to bot with Open-AI machine, it gets question and send it to the “get\_completion”. It is also catches possible errors, so there is little chance for program to be terminated.

“get\_completion” send a request and then get a response using open-ai library

Lines of code are explained in the code by comments.



### Links

<https://platform.openai.com/docs/api-reference/chat/object>

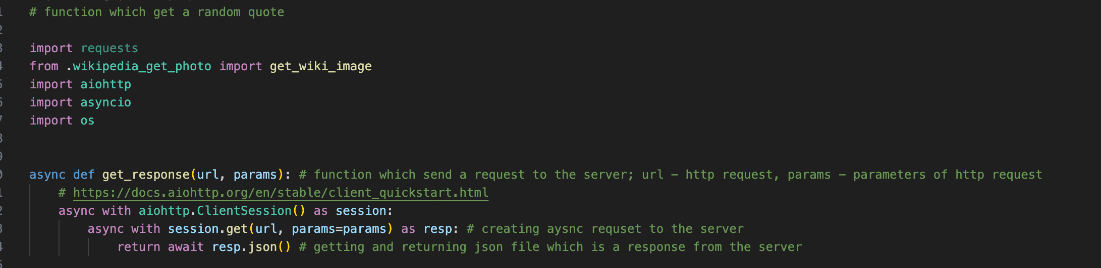
<https://platform.openai.com/docs/guides/gpt/chat-completions-api>

<https://blog.enterprisedna.co/how-to-use-chatgpt-for-python/>

<https://platform.openai.com/docs/plugins/examples>

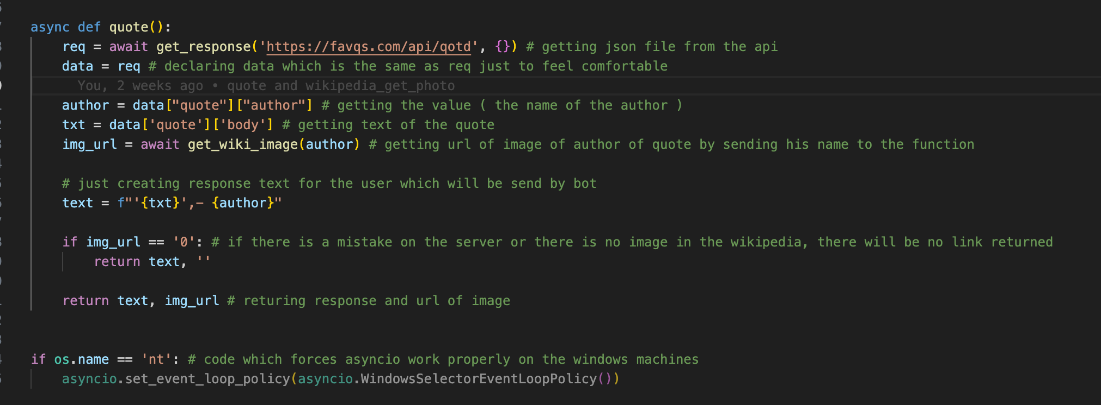
-------------------------------------------------------------------------------------------------------------------------------------

## Quote



First, I import the function “get\_wiki\_image” from “wikipedia\_get\_photo”. This function returns the URL of the photo of the author of the quote. Then I import libraries which are necessary for proper working of my bot.

Function “get\_response” takes “url,” which is link of the API request, and “params” which will be empty as there are no specific parameters of the request. Then this function, using specific async library, send a request to the server and get a response with JSON file, which will be returned to the main function.



Function “quote” is the main one in the file. It calls “get\_response” function to get json file with all needed data, then duplicate it to the “data” value. Then we get author and text of the photo from the json file.



(An example of the JSON file)

After this I call another function (“get\_wiki\_image” ) which will find the url of image of author.

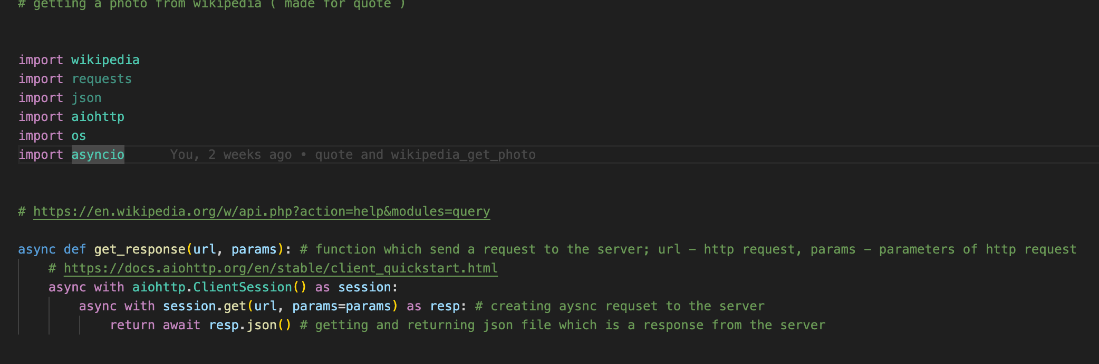
Then I create the text which will be used as a response to the user in TG.

### Links

<https://favqs.com/api>

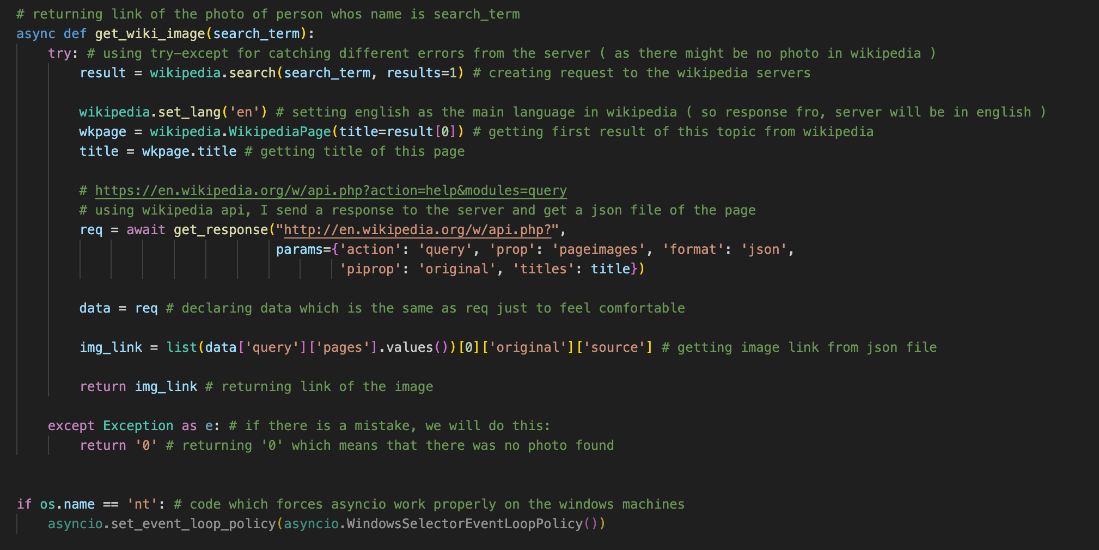
-------------------------------------------------------------------------------------------------------------------------------------

## Wikipedia



Firstly, I import necessary libraries. “wikipedia” library is used to work with wikipedia pages and last three are used to make program work with async style of programming.

There is “get\_response” function which has been already described above.



In function “get\_wiki\_image,” “search\_term” is the name of the author.

I use TRY-EXCEPT to make sure that there are no mistakes while trying to find page (if there is no page, or there is server error), if there is a mistake, the return will be 0, which in this case ,means nothing.

The idea of the function is to use wikipedia library to find specific page and get a title of it. Then, I need to use API which will collect all the data from this page and return to me a json file. After this, I take URL of the photo from the file and return it.

### Links

<https://en.wikipedia.org/w/api.php?action=help&modules=query>

<https://www.educative.io/answers/how-to-get-all-the-image-urls-from-a-wikipedia-page-using-python>

<https://www.geeksforgeeks.org/how-to-extract-wikipedia-data-in-python/>

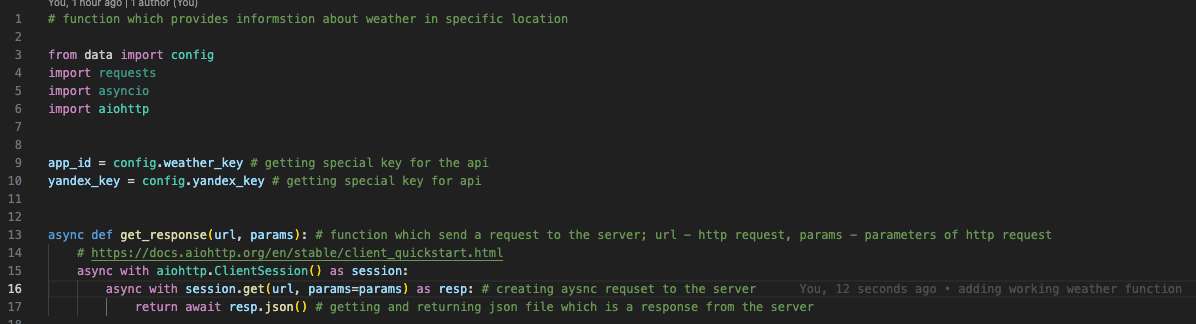
<https://pypi.org/project/Wikipedia-API/>

<https://stackoverflow.com/questions/8363531/accessing-main-picture-of-wikipedia-page-by-api>

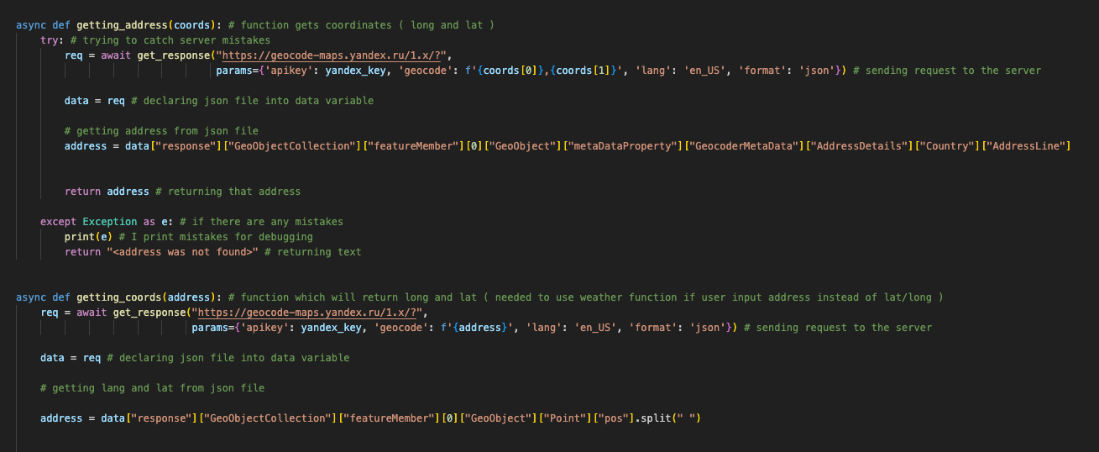
<https://docs.aiohttp.org/en/stable/client_quickstart.html>

-------------------------------------------------------------------------------------------------------------------------------------

## Weather



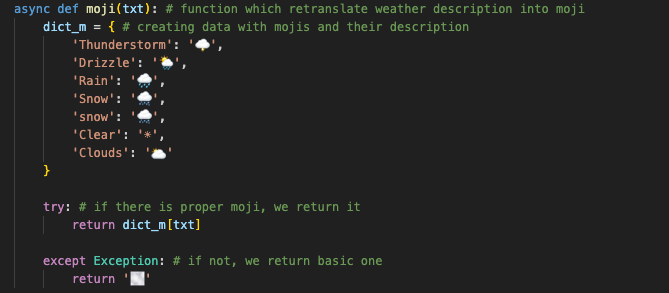
First, as it has been already done before in other functions, I import extensions which will allow my program works properly and then write “get\_response” function.



There are two functions which are connected to the Yandex API ( <https://yandex.com/dev/> ). They are needed to work with user location.

“getting\_coords” is needed to get longitude and latitude from user’s input. Function gets address and then, using Yandex API, finds longitude and latitude.

“getting\_address” is remarkably like the function above, but it reflects it – it gets longitude and latitude and then finds the address.



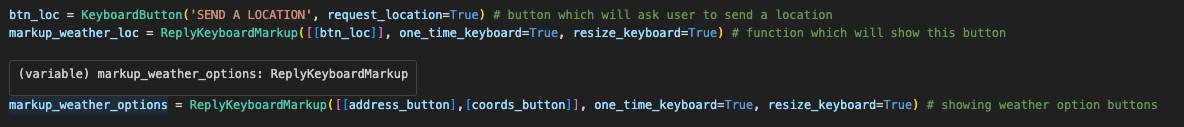
There is also “moji” function which was made to make output less formal.



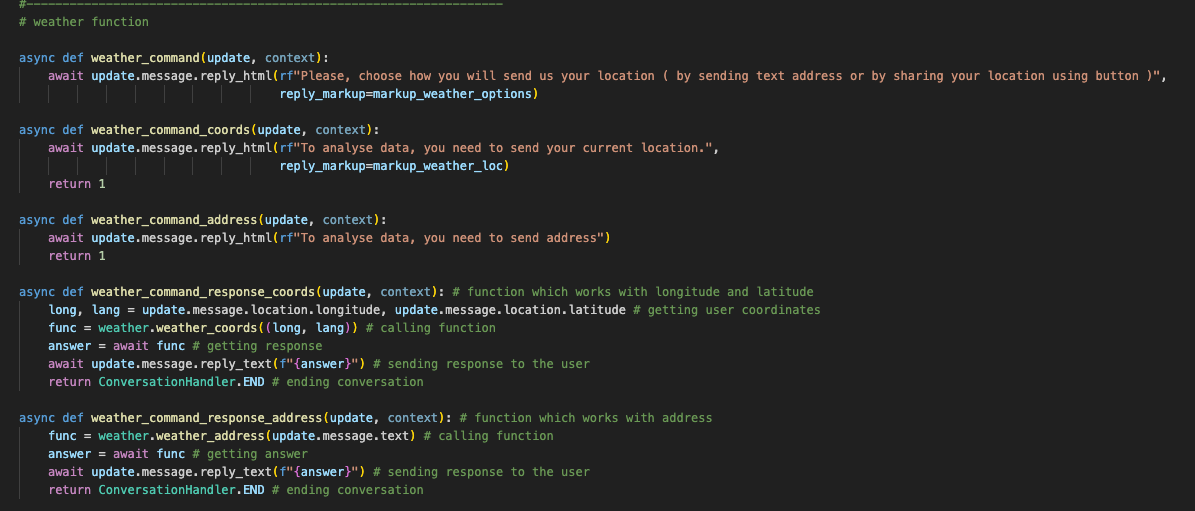
There are two main functions in the code which provide weather information.

“weather\_coords” gets longitude and latitude and then goes to the OpenWeatherMap server with a request. Then, after getting response, program analyses json file with the response and takes needed data. Then it creates a response to the user and in the end sends it.

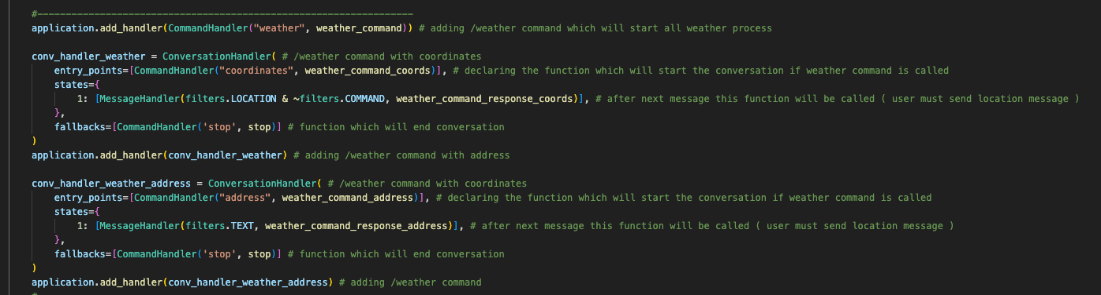
“weather\_address” works in the same way, but instead of getting longitude and latitude, it gets address and then, using “getting\_coords” function, it gets longitude and latitude.



In the main.py I create buttons which will help user to work with this function. One will be connected to the location sending by using innate GPS tracker. And another two buttons will be just choices of how to send location.



There are 5 functions which are all used in conversation.



Conversation starts with “weather” command, it asks user how he wants to send location. If user chooses coordinates, the program will ask him a permission to send his location which is recorded by phone. If user chooses address, he will need to input address in a message and send it to bot. By the end of this manipulations, bot sends information about weather in chosen location.

### Links

<https://yandex.ru/dev/weather/doc/dg/concepts/about.html>

<https://openweathermap.org/>

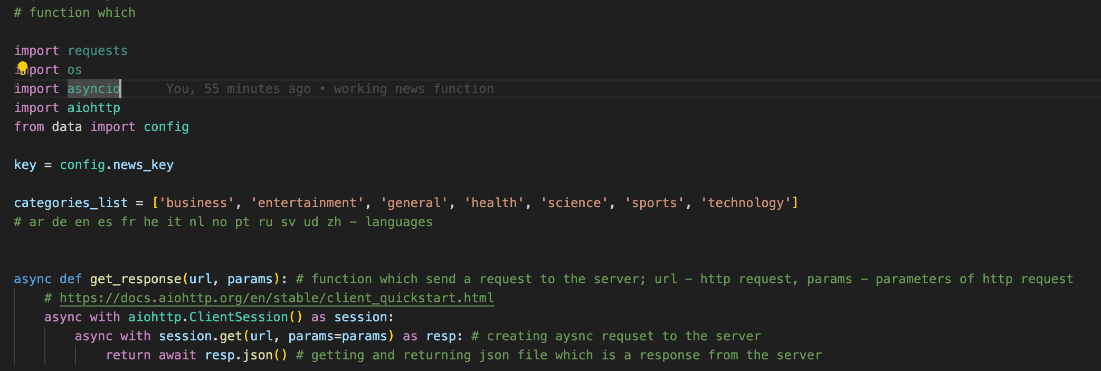
<https://openweathermap.org/current>

<https://yandex.com/dev/maps/geocoder/>

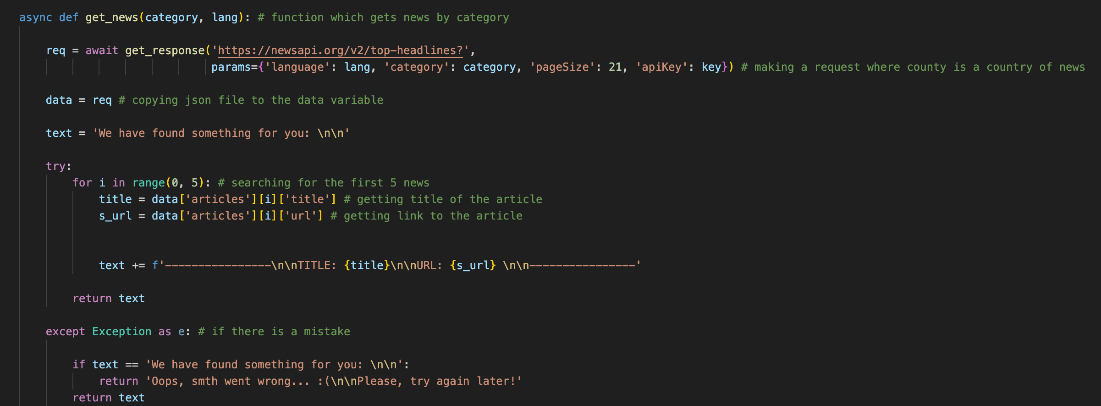
<https://yandex.com/dev/geocode/doc/en/request>

-------------------------------------------------------------------------------------------------------------------------------------

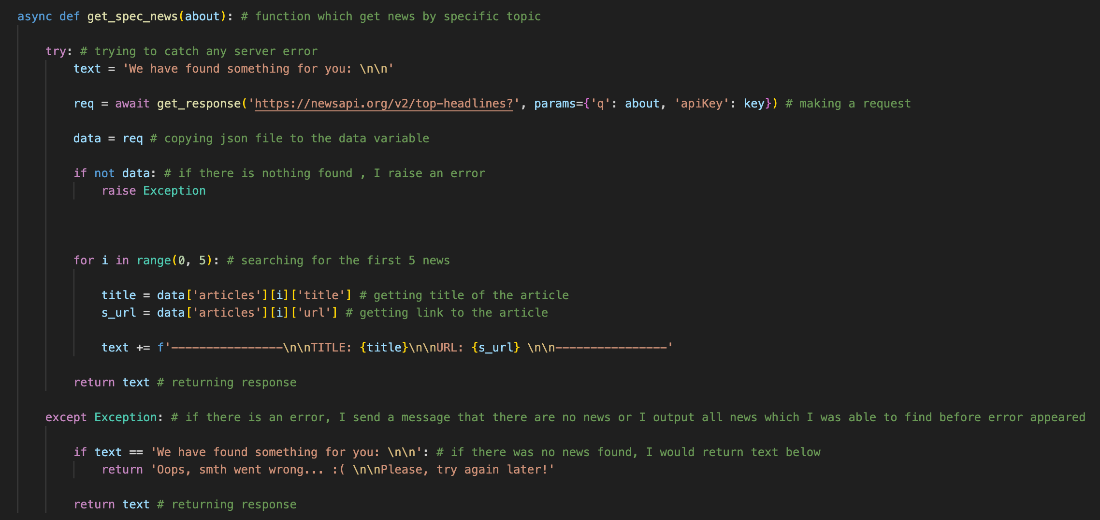
## News



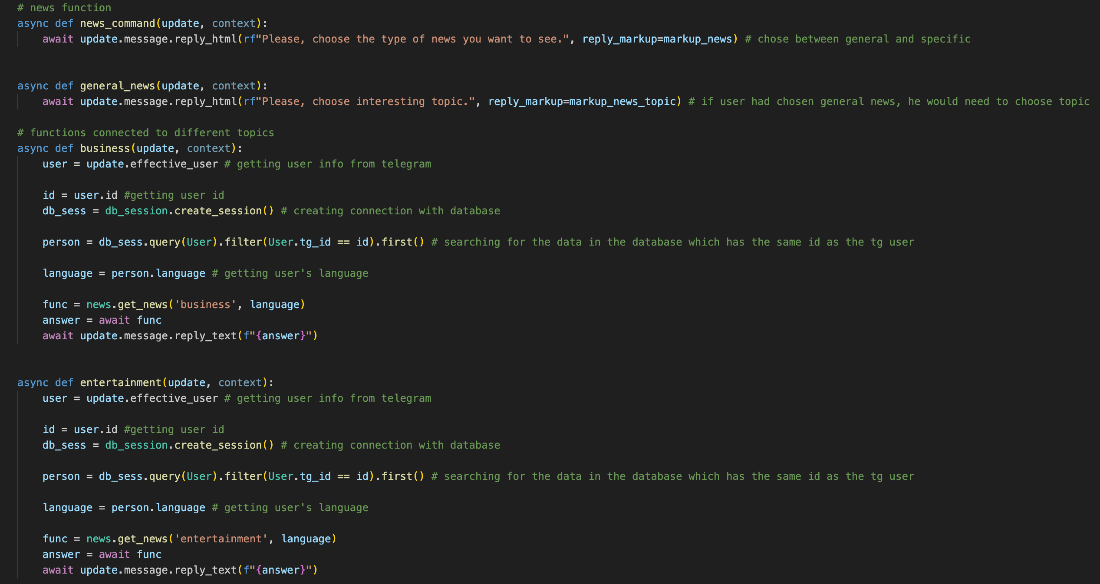
Same procedure as in the previous functions.



Function “get\_news” gets category, which is one of the topics of news, and language, which is language set up by user (default – English). It sends a request to the server, and then gets a response with 21 news. Then it sends a text with title and link to this news to the user.



The same idea is in the “get\_spec\_news” which gets only topic by user. Then it tries to find related to this topic news and in the end sends text with titles and URLs of the news.



All general news is built in the same way – they get language from database, then call the function, and send a response from this function to user.

Specific news (where user must write what he wants to find) are built as a conversation function (the same as it was with gpt function).

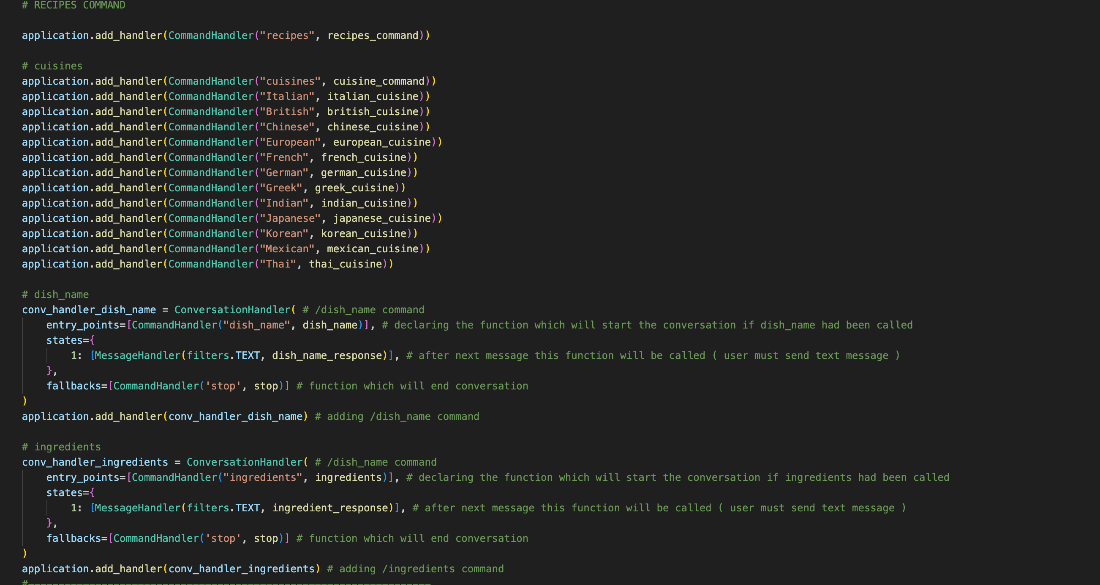
### Links

[https://newsapi.org](https://newsapi.org/)

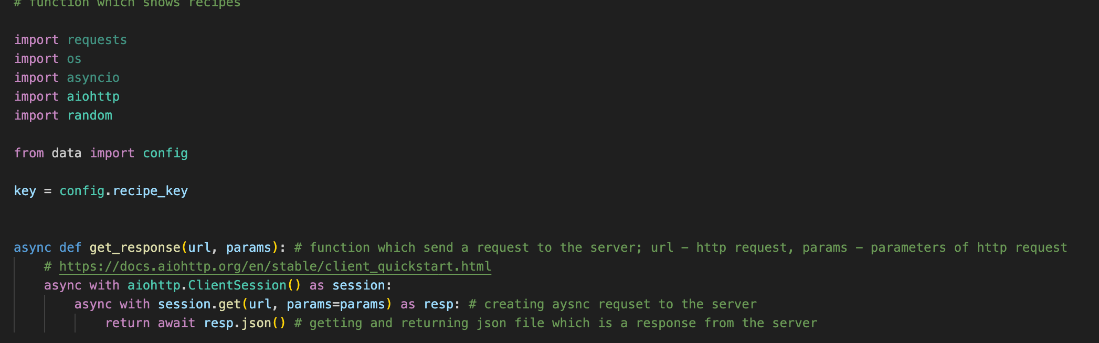
<https://newsapi.org/docs/endpoints/>

-------------------------------------------------------------------------------------------------------------------------------------

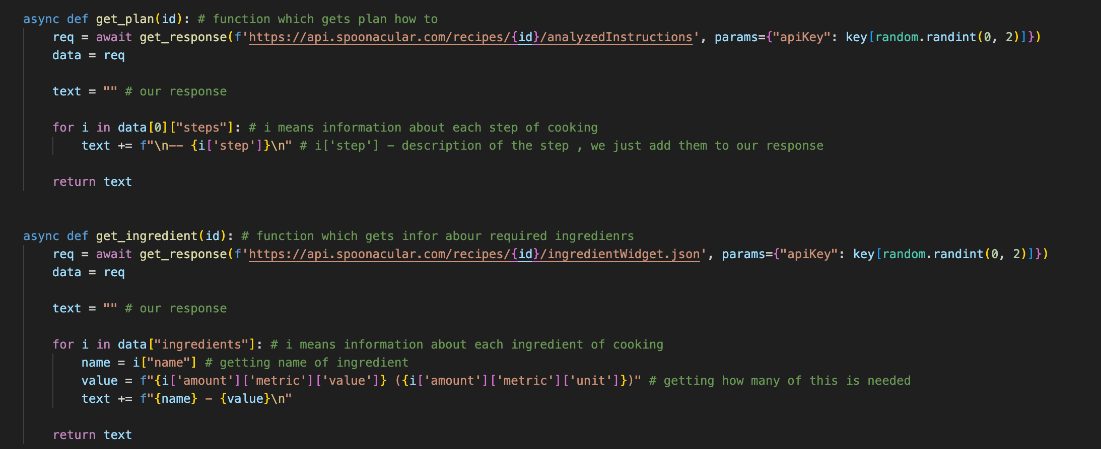
## Recipes



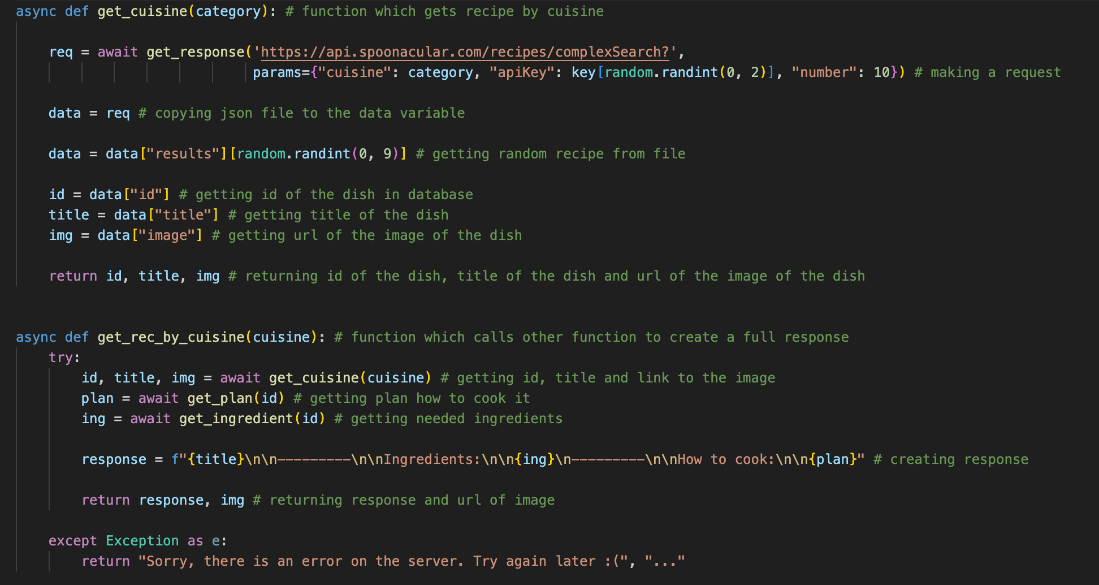
In the main file I declare all the functions as usual, there are 2 conversation handlers which help me to work with “dish\_name” function and “ingredients” function.



There is well-known get\_response function which is being the same in every file.

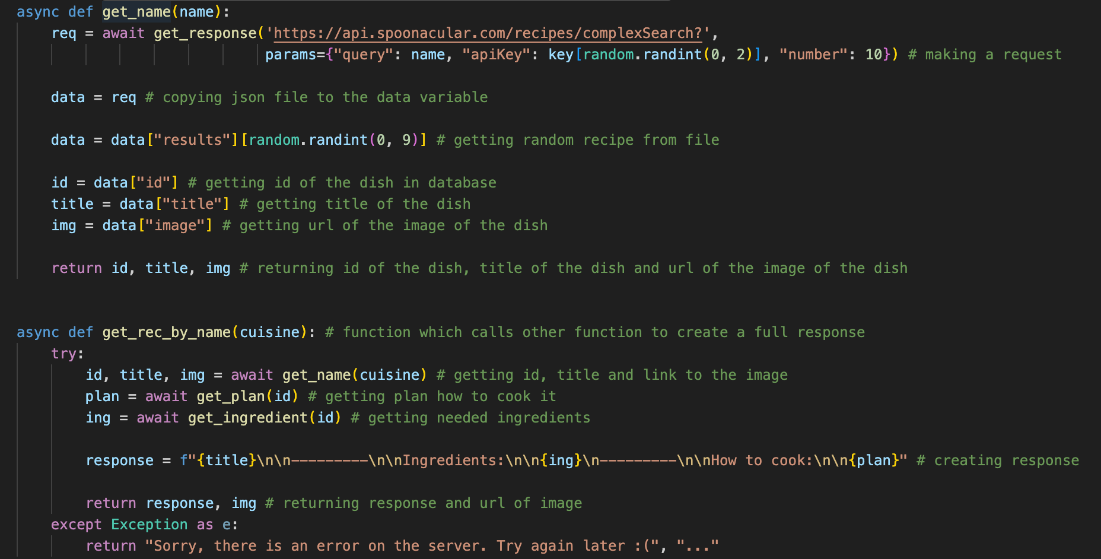


These two functions are needed to get plan how to cook dish and what ingredients are needed to do this. They both get ID and using it find this information using API.

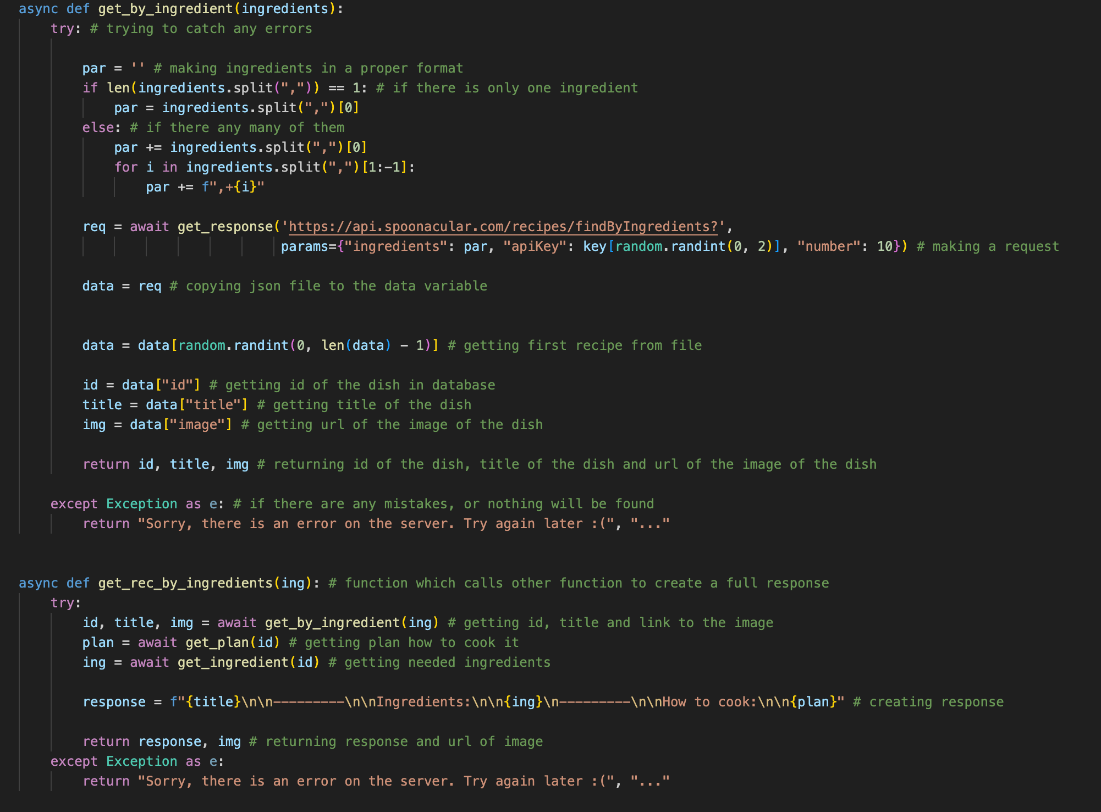


“get\_cuisine” function is needed to find recipe by the cuisine type, for example, if user wants to eat something from Italian menu, he chooses this type of cuisine and then script tries to find any dish from the required cuisine.

“get\_rec\_by\_cuisine” function makes a response for the user, it gets id of the dish and then goes to another two functions and gets information from them, then it summarizes it and sends to user.



These functions have the same idea as functions above, but they find dish by its name. For example, if user wants ‘pasta’, he will input pasta and bot will get the dish connected to pasta.



These functions find dish by ingredients.

### Links

<https://spoonacular.com/food-api/docs#Get-Analyzed-Recipe-Instructions>

<https://spoonacular.com/food-api/docs#Cuisines>

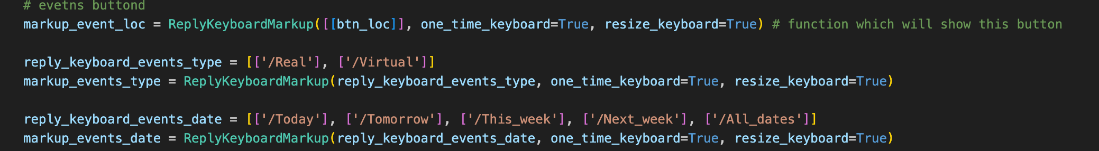
<https://spoonacular.com/food-api/docs#Ingredients-by-ID>

<https://spoonacular.com/food-api/docs#Search-Recipes-Complex>

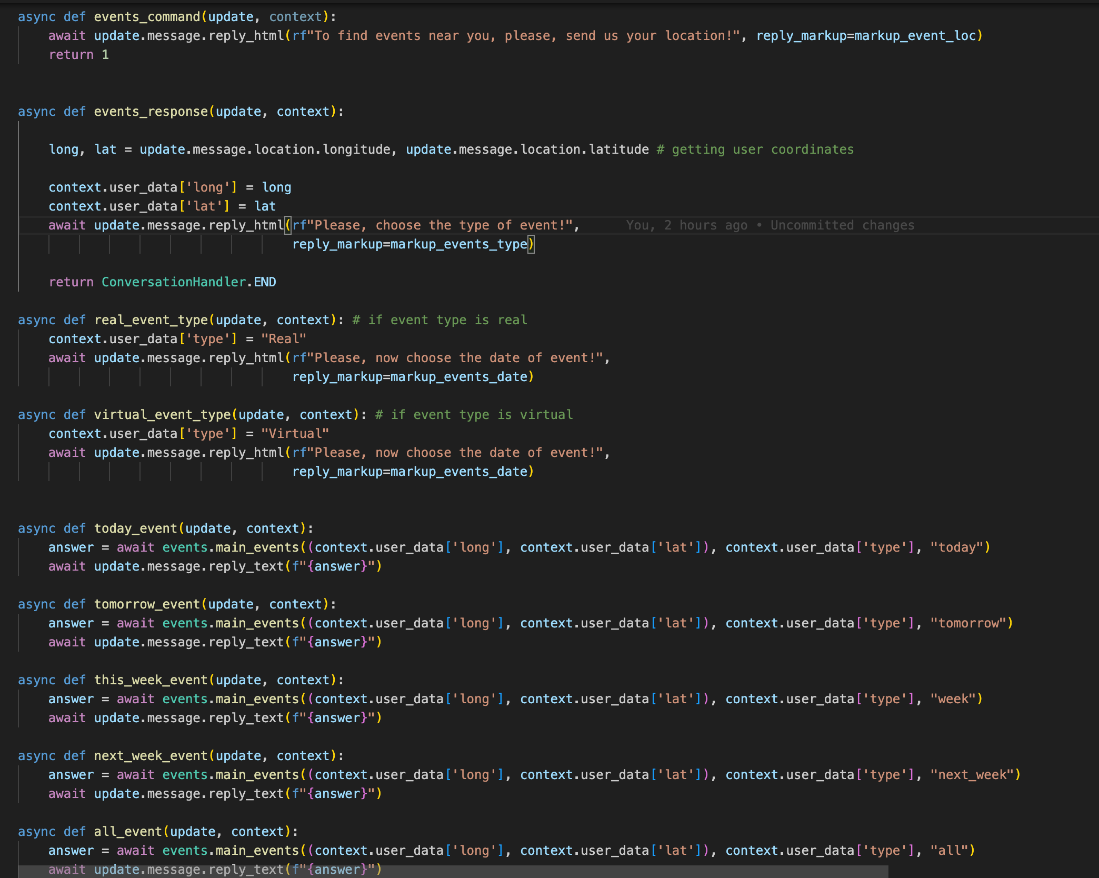
<https://spoonacular.com/food-api/docs#Search-Recipes-by-Ingredients>

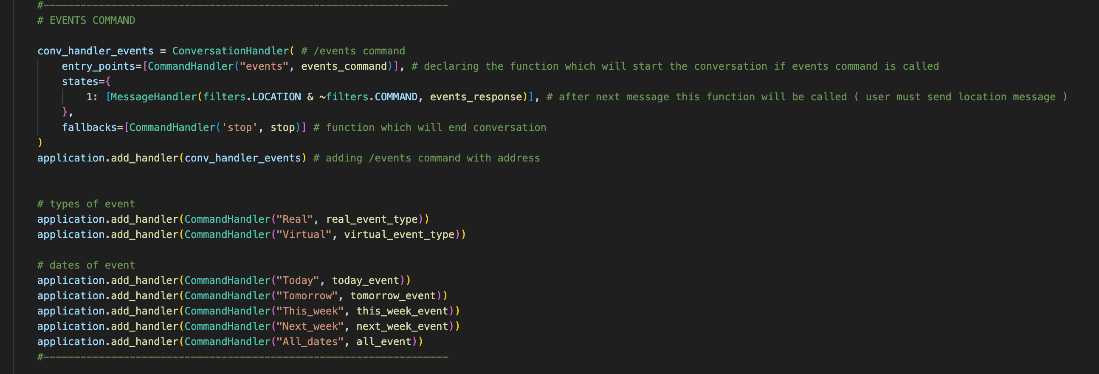
-------------------------------------------------------------------------------------------------------------------------------------

## Events

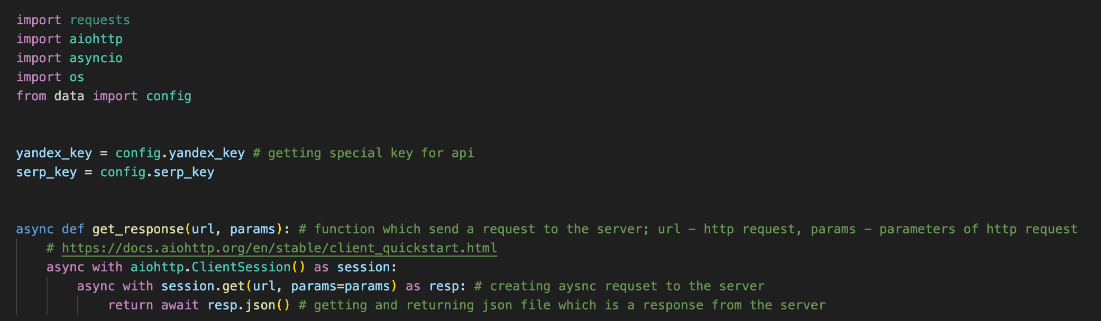


In the main code I declare buttons which will be helpful for the taking user’s answers.

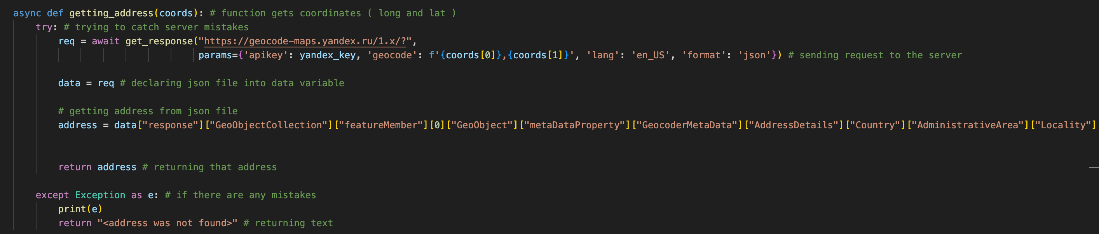




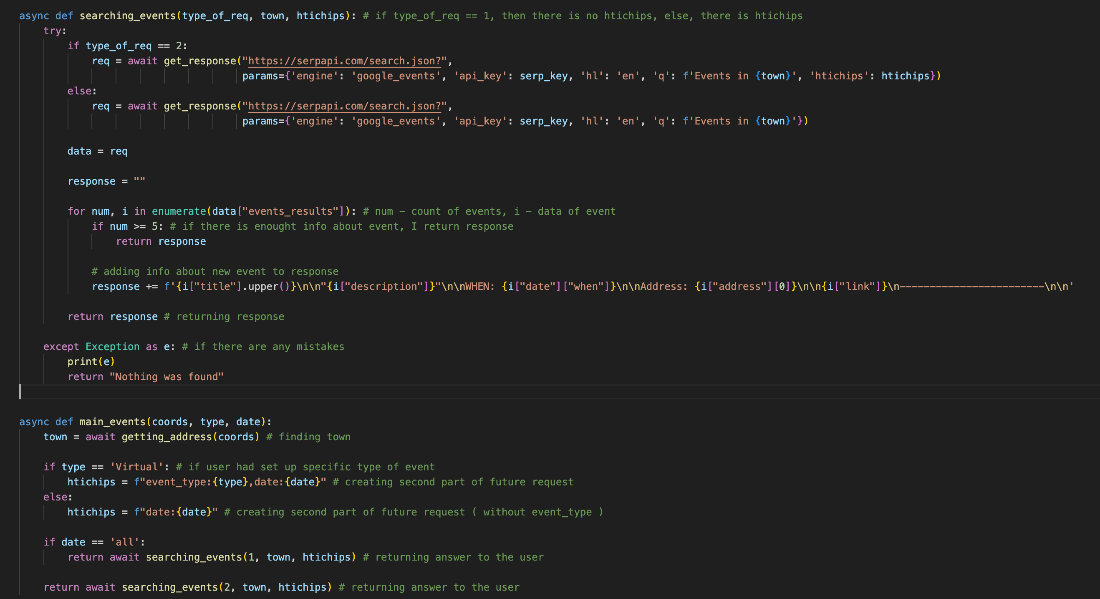
After, in the main file I write functions which asks questions to the user, and then in the conversation form they summarise all the answers in the dictionary and then call the function which will return the response.



In the events.py there is as always “get\_respones” function.



“getting\_address” function returns town by analysing coordinates of the user. It is important function as the main request includes name of the town.



“searching\_events” makes a request to the server and creates a response.

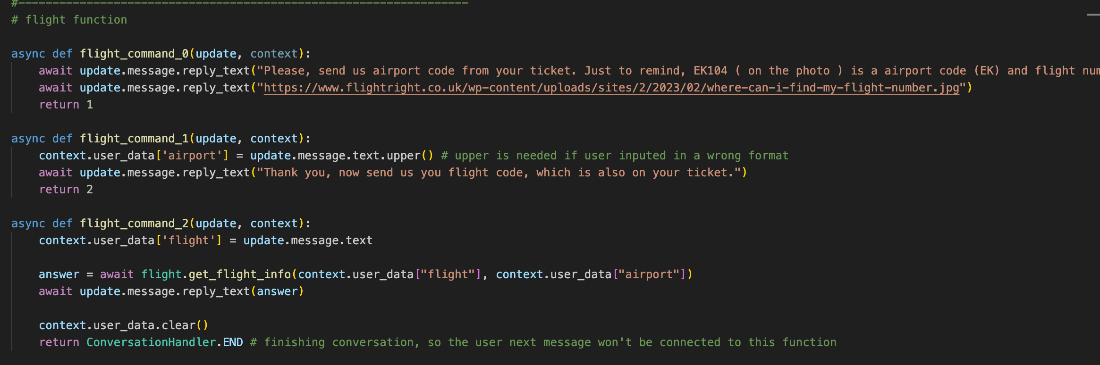
“main\_event” is needed only to connect all these two functions and then return final response t

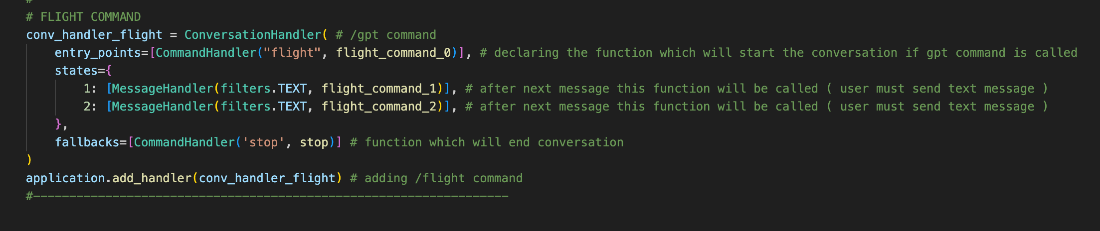
### Links

<https://serpapi.com/google-events-api>

-------------------------------------------------------------------------------------------------------------------------------------

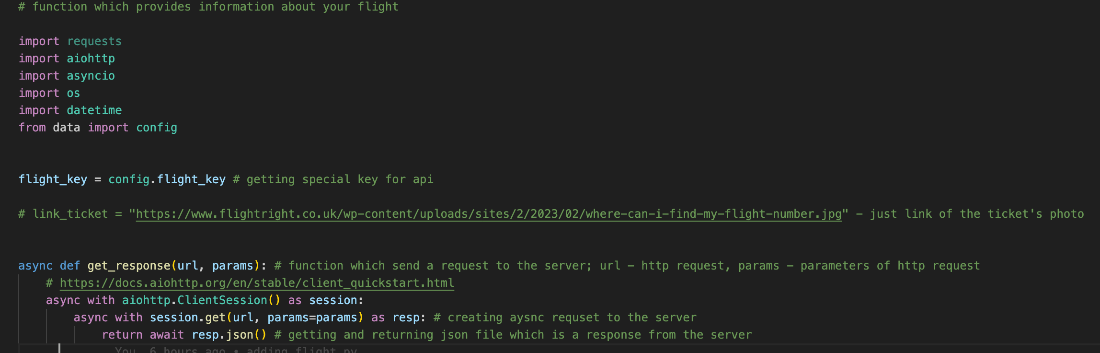
## Flight



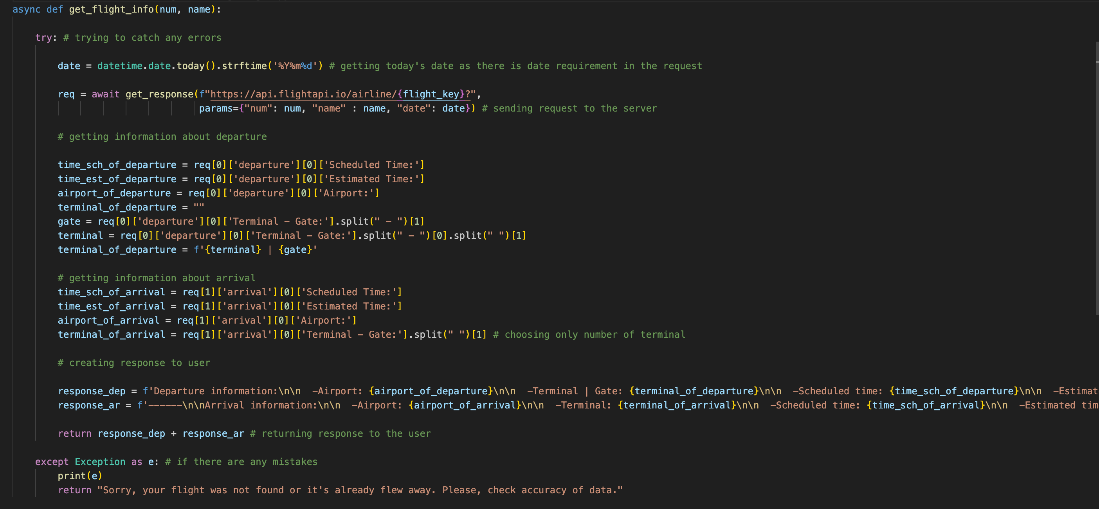


In the main code function was declared in a straightforward way – it is 3 step conversation with the user.

In the first function I ask user for the airport code and show program that it need to move to the second function. In the second function I store user’s answer in the database and ask question about flight number and show program that it need to call third function. And finally, in the third function I get user’s response and call function from the “fligh” file which will find information about user’s flight.



As always there are imports and basic “get\_response” function.



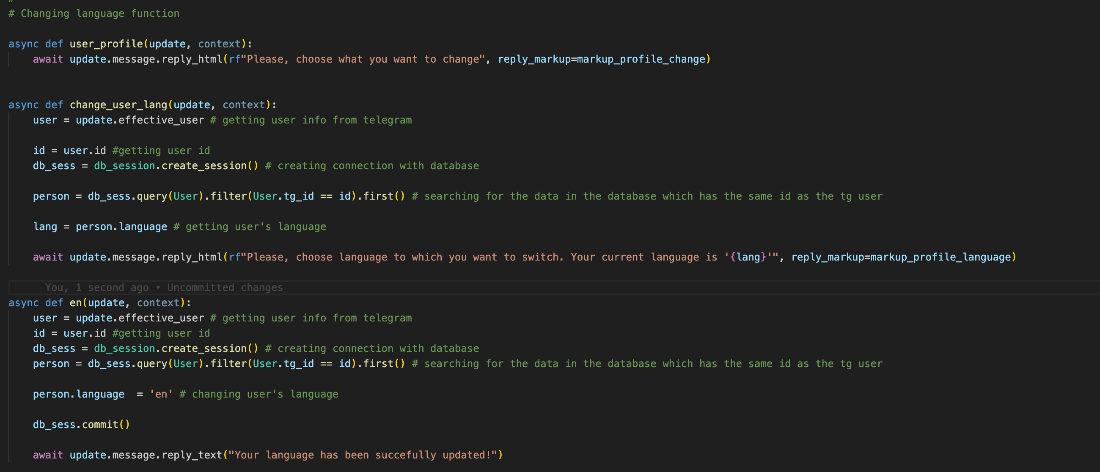
This is the main function of the file which returns information about the flight. It gets as parameters num (flight number) and name (airport code). Then it goes to the server with these parameters (date of the flight is always today’s date) and after this I analyse json file from the response to get needed information. Then I summarise all the data and return response to the user.

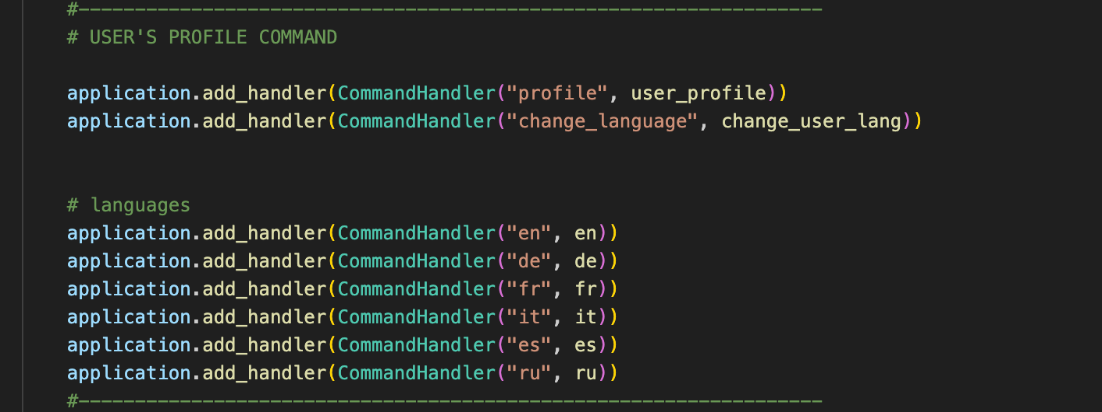
### Links

<https://docs.flightapi.io>

-------------------------------------------------------------------------------------------------------------------------------------

## User’s profile





This function is needed to change different settings in the bot. It gets current language of the user from database and ask him if he want to change it. After pushing buttons language will be updated