Run OpenAI’s ChatGPT Model in C++

***Note:*** *This File is created by and a property of* ***NeuroCode*** *YouTube Channel*

**YouTube:** https://www.youtube.com/@NeuroCoode

1. Install Git if not installed already
   1. <https://git-scm.com/downloads>
2. Open a terminal (Command Prompt, PowerShell, or Git Bash) and clone the vcpkg repository from GitHub:

git clone <https://github.com/microsoft/vcpkg.git>

1. Navigate to the vcpkg directory  
   *cd vcpkg*
2. Run the bootstrap script to build vcpkg:  
   *.\bootstrap-vcpkg.bat*
3. Integrate vcpkg with your environment:  
   .\vcpkg integrate install
4. NOW INSTALL LIBCURL and jsoncpp libraries
5. Use vcpkg to install libcurl: ( in the same vcpkg directory)  
    .\vcpkg install curl
6. Install jsoncpp (optional, but often used with libcurl for handling JSON data):  
    .\vcpkg install jsoncpp
7. Create C++ Project   
   Then we will add paths to our CodeBlocks  
     
   PROJECT > BUILD OPTIONS > SEARCH DIRECTORIES
8. Include the path for vcpkg in search directory  
   A screenshot of a computer

   Description automatically generated
9. Under the Link libraries tab, add the necessary libraries:  
   Add libcurl.lib and jsoncpp.lib (if you installed jsoncpp).  
   A screenshot of a computer

   Description automatically generated
10. Add Json.hpp file in your C++ Project directory

git clone <https://github.com/nlohmann/json.git>

* 1. copy the json.hpp file from the single\_include/nlohmann directory to your project directory.

1. Code

#include <iostream>

#include <string>

#include <curl/curl.h>

#include "json.hpp"

using json = nlohmann::json;

using namespace std;

static size\_t WriteCallback(void\* contents, size\_t size, size\_t nmemb, void\* userp)

{

((string\*)userp)->append((char\*)contents, size \* nmemb);

return size \* nmemb;

}

string getChatGPTResponse(const std::string& prompt)

{

CURL\* curl;

CURLcode res;

string readBuffer;

curl\_global\_init(CURL\_GLOBAL\_DEFAULT);

curl = curl\_easy\_init();

if (curl) {

struct curl\_slist\* headers = NULL;

headers = curl\_slist\_append(headers, "Content-Type: application/json");

headers = curl\_slist\_append(headers, "Authorization: Bearer YOUR-API-KEY");

curl\_easy\_setopt(curl, CURLOPT\_URL, "https://api.openai.com/v1/chat/completions");

curl\_easy\_setopt(curl, CURLOPT\_HTTPHEADER, headers);

string jsonPayload = R"({

"model": "gpt-3.5-turbo-1106",

"messages": [

{"role": "system", "content": "You are a helpful assistant."},

{"role": "user", "content": ")" + prompt + R"("}

],

"max\_tokens": 15

})";

curl\_easy\_setopt(curl, CURLOPT\_POSTFIELDS, jsonPayload.c\_str());

curl\_easy\_setopt(curl, CURLOPT\_WRITEFUNCTION, WriteCallback);

curl\_easy\_setopt(curl, CURLOPT\_WRITEDATA, &readBuffer);

res = curl\_easy\_perform(curl);

if (res != CURLE\_OK) {

fprintf(stderr, "curl\_easy\_perform() failed: %s\n", curl\_easy\_strerror(res));

} else {

try {

auto response\_json = json::parse(readBuffer);

return response\_json["choices"][0]["message"]["content"];

} catch (json::parse\_error& e) {

cerr << "JSON parsing error: " << e.what() << std::endl;

} catch (json::type\_error& e) {

cerr << "JSON type error: " << e.what() << std::endl;

} catch (std::exception& e) {

cerr << "Error: " << e.what() << std::endl;

}

}

curl\_easy\_cleanup(curl);

}

curl\_global\_cleanup();

return "";

}

int main()

{

string prompt;

cout << "Enter your prompt: ";

getline(std::cin, prompt);

do {

string response = getChatGPTResponse(prompt);

cout << "ChatGPT response: " << response << std::endl;

cout << "Enter your prompt (0 to terminate): ";

getline(cin, prompt);

} while (prompt!="0");

return 0;

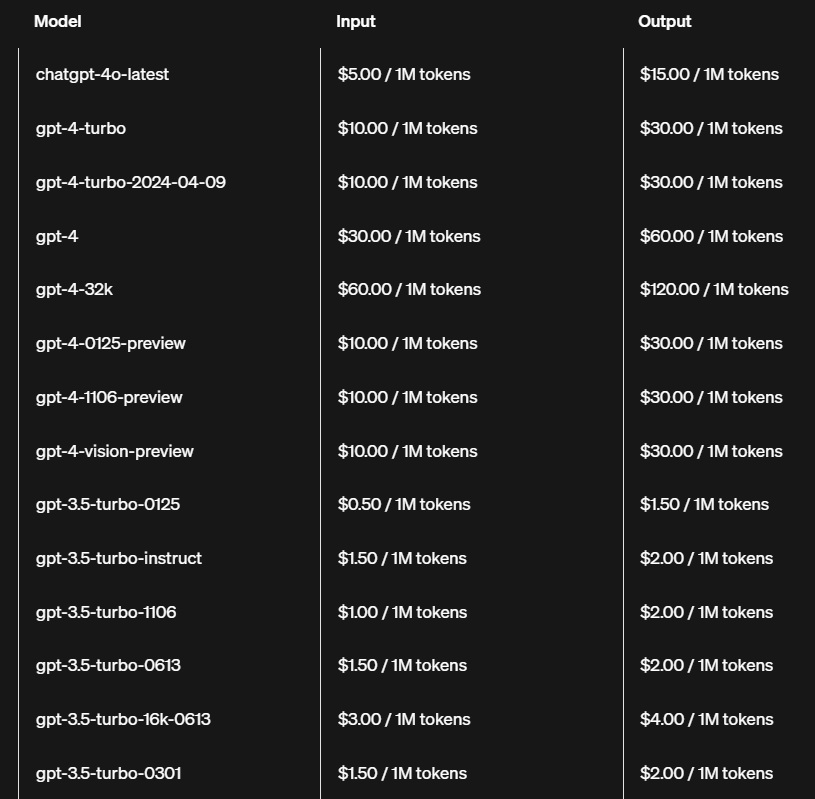
}

1. Add API KEY

Link for Generating API Key

* 1. https://platform.openai.com/api-keys

1. Choose a model



Source: [Models - OpenAI API](https://platform.openai.com/docs/models)