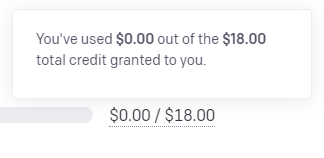
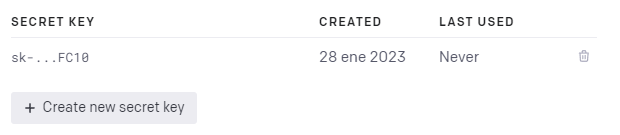
**# # # Alexa Integration with OpenAI (+Costs)**

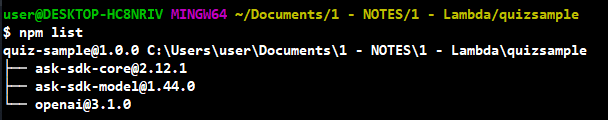
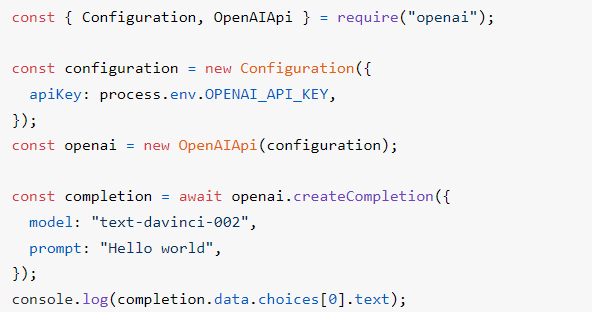
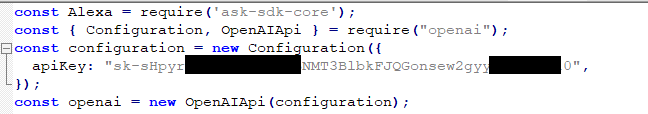
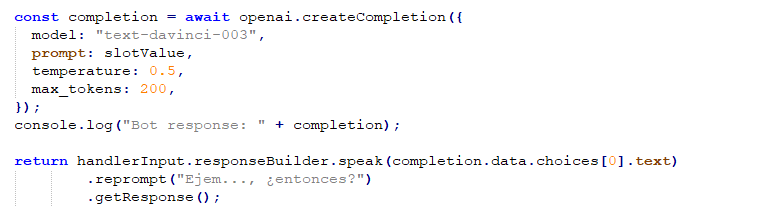
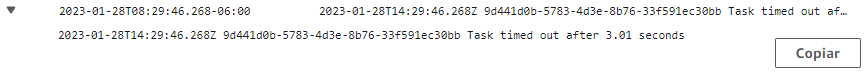
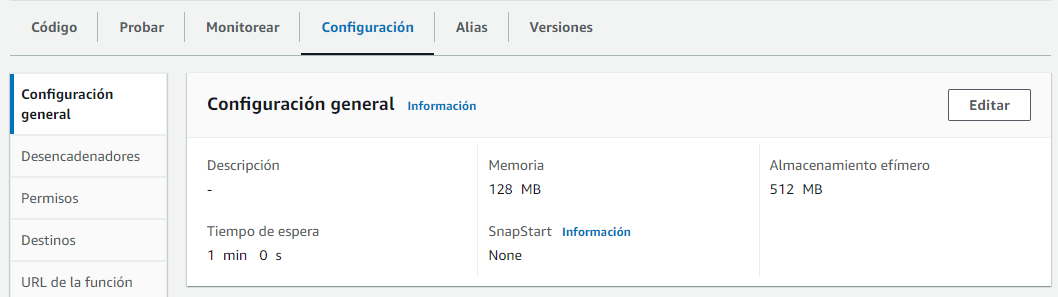
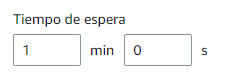
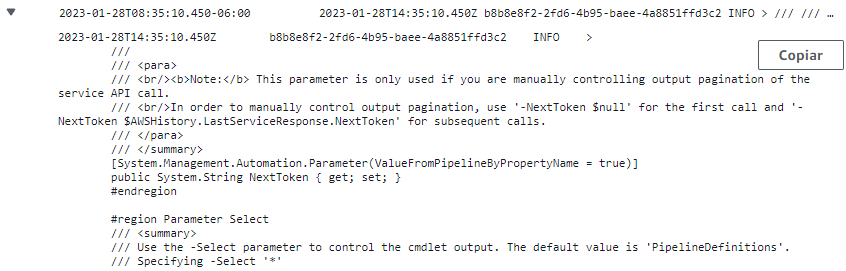
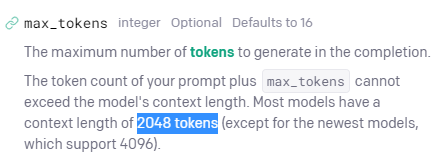
I have 18 dollars of free trial that expires on “*1 de mayo de 2023”* just for creating an Open AI account.  
  
\* <https://beta.openai.com/account/usage>

**Making my first OpenAI request:**

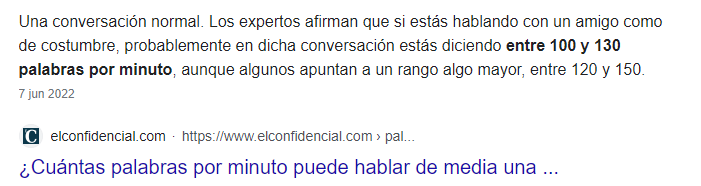
* I’ll create a secret key to make HTTP requests.  
    
  \* <https://beta.openai.com/account/api-keys>
* One important setting is called **temperature**…
  + Temperature is a value between 0 and 1 that essentially lets you control how confident the model should be when making predictions.  
    *\* 0 will always retrieve the same answer, and* ***as closer to 1 the model will retrieve more diversity of answers****.*
* **My first request:**
  + curl https://api.openai.com/v1/completions -H "Content-Type: application/json" -H "Authorization: Bearer ***<SECRET-APY-KEY>***" -d '{"model": "text-davinci-003", "prompt": "*Dime para que sirve el asistente Alexa.*", "temperature": 0, "max\_tokens": 50}'  
    **\* Response:**
    - {"id":"cmpl-5dg3CPpdrPdRyxbKNB5v462qr2F","object":"text\_completion","created":1674914454,"model":"text-davinci-003","choices":[{"text":"\n\*nAlexa es un asistente virtual de Amazon que te ayuda a realizar tareas cotidianas como reproducir música, responder preguntas, controlar dispositivos*","index":0,"logprobs":null,"finish\_reason":"length"}],"usage":{"prompt\_tokens":12,"completion\_tokens":50,"total\_tokens":62}}

**Notes about OpenAI NodeJS library:**

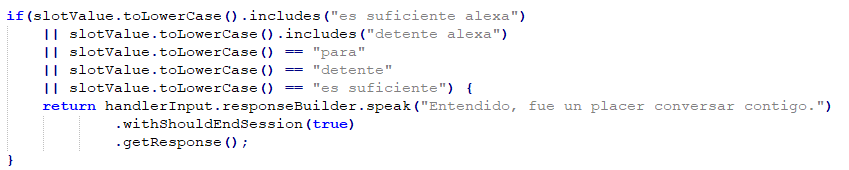
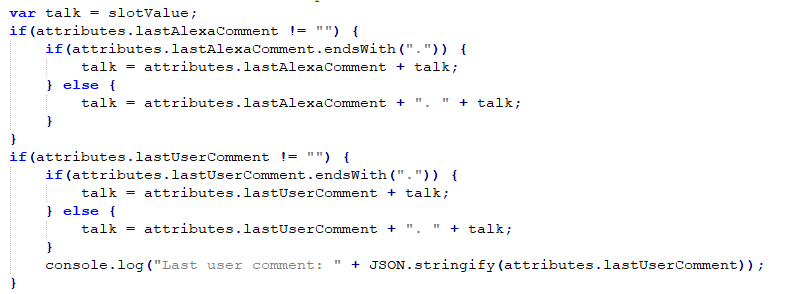
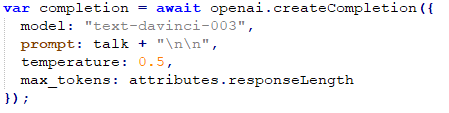
\* <https://beta.openai.com/docs/api-reference/introduction>

* $ npm install openai  
    
  \* <https://stackoverflow.com/questions/10972176/find-the-version-of-an-installed-npm-package>
* Library usage:  
  \* <https://www.npmjs.com/package/openai>, <https://beta.openai.com/docs/libraries/node-js-library>  
  
* I added the configuration to the imports section:  
  
* Then I declared as *async* the handler method:  
  
* And then I made the request from inside the handler…  
  
* **\* Error:** “*Task timed out after 3.01 seconds*”  
    
  **\* Solution:**
  + I changed the timeout for the AWS Lambda function in “*General Configuration*”.  
      
    *\* Click on “Edit” and...*  
    \* <https://docs.aws.amazon.com/lambda/latest/dg/configuration-function-common.html#configuration-timeout-console>
* To keep the context of the conversation it seems it is required to include the previous interactions in the prompt.  
  \* <https://community.openai.com/t/how-do-you-maintain-historical-context-in-repeat-api-calls/34395/8>
* **\* Error:** with “para” (“stop”) slot value…  
    
  **\* Solution:** Try..catch block (<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/try...catch>).
* The maximum number of tokens that the model supports is 2048 tokens.  
    
  \* <https://beta.openai.com/docs/api-reference/completions/create#completions/create-max_tokens>

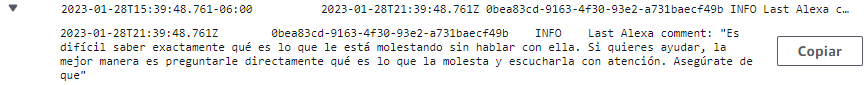
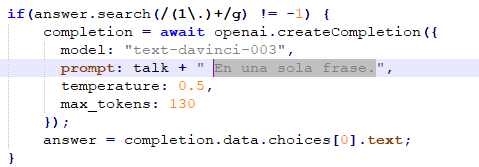
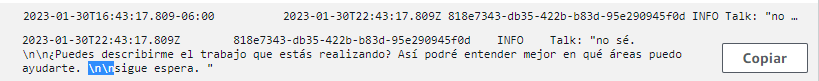
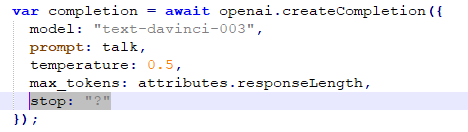
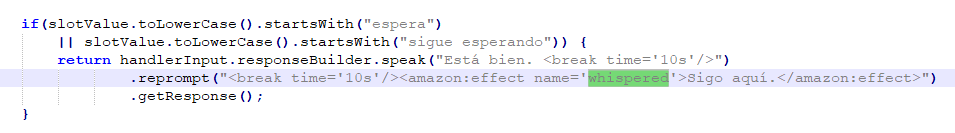
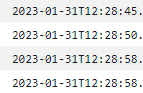
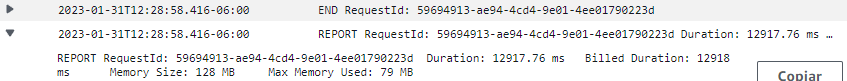
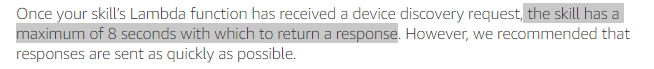
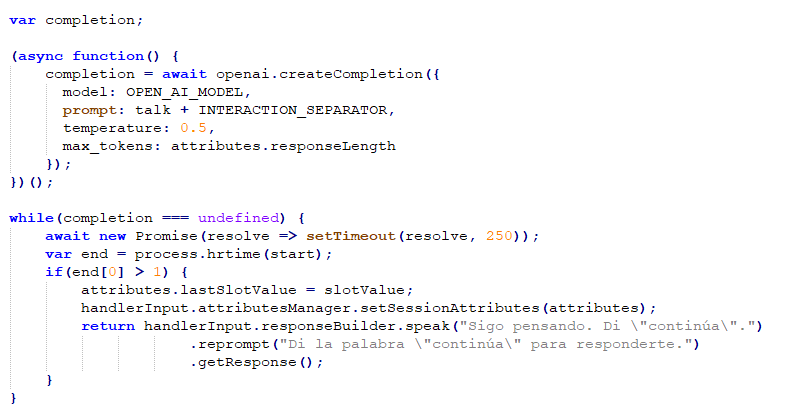
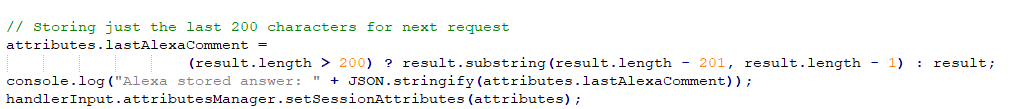
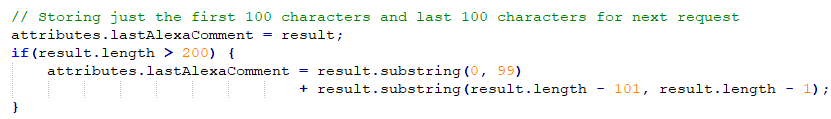
**About costs:**

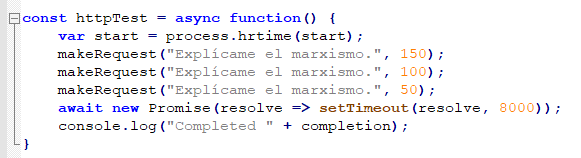
* It seems an regular person says max 300 words per minute.  
  
  + Every interaction will cost 400 tokens, or approx. $0.01 US dollars. Considering an interaction like 200-300 words, which isn’t likely.

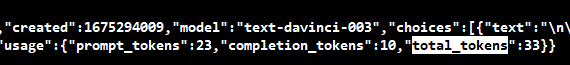
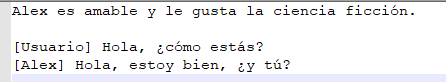
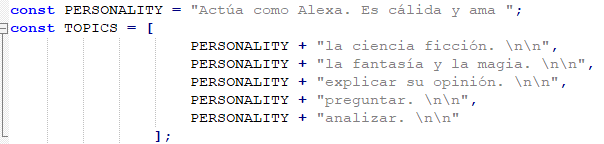
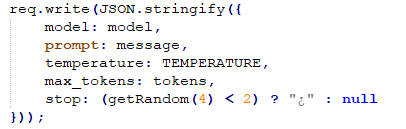
**Integrating Open AI to Alexa:**

* I was able to integrate the Davinci AI (completion feature) inside my Alexa Skill.
* I added stop commands…  
  
* Last user input and Alexa response is considered.  
  
* The AI configuration:  
  

**Challenges:**

* How can I deal with **short/cut answers**? **R.** Saying “Continue”.  
  \* <https://community.openai.com/t/completion-cut-off-and-unknown-characters/16708/2>, <https://www.reddit.com/r/OpenAI/comments/zgkulg/chatgpt_often_will_not_finish_its_code_or/>
* It usually finishes without completing a phrase:  
    
  **\* Solution:** I’ll remove the last sentence when this happens.  
  
* To **deal with lists**, if I detect one I ask again for the answer adding a modifier “*in one phrase*”, giving more tokens to retrieve the answer.  
  
* **Alexa never adds question marks** to the slot value, but this seems to not be a problem for the AI to retrieve coherent answers.
* In order to make the AI recognize the text to complete as a conversation, I added “\n\n” at the end of each interaction (from user or from AI).  
  
* To make it feel more as a conversation, I did the AI stop when a question mark is reached.  
    
  **\* This didn’t work** because the stop sequence isn’t included in the retrieved response from the AI, I decided to cut the response once it comes back using “*.search(/\?/g)”*.
* To make Alexa whisper:  
    
  \* <https://developer.amazon.com/en-US/docs/alexa/custom-skills/speech-synthesis-markup-language-ssml-reference.html#amazon-effect>
* **Multi-language problem…**, there’s no way to recognize if something has to be pronounced in English or Spanish.  
  \* <https://us-east-2.console.aws.amazon.com/cloudwatch/home?region=us-east-2#logsV2:log-groups/log-group/$252Faws$252Flambda$252FQuizSample/log-events/2023$252F01$252F30$252F$255B$2524LATEST$255D1f30fbcdcb754640b084b9838c55bf57$3Fstart$3D1675118263439$26refEventId$3D37356385572588309253190015140528546342398399410461343753>
* It seems when the response takes more than 10 seconds Alexa interrupts the connection. I’m making two requests, the first request took 5 seconds with 80 tokens, and the second one 8 seconds with 150 tokens approx.  
    
    
  \* <https://amazon.developer.forums.answerhub.com/questions/205700/waitcontinue-in-alexa-skill-response.html>
  + I can make the first request with 10 tokens just to know if the response is a list or not.
* The completion time of one request is more than 10 seconds when the talk reaches certain length, in this scenario **the** **Alexa Skill closes the connection** with AWS Lambda.  
    
  \* <https://developer.amazon.com/en-US/blogs/alexa/device-makers/2019/04/4-tips-for-implementing-device-discovery-in-your-smart-home-skills>  
  **\* Attempts:**
  + I tried to use an async function but it seems the AWS Lambda function won’t retrieve a response until all the threads (Promises) complete their execution, and they cannot be cancelled.  
    
  + I have to cut the response stored, so that   
    
  + **\* Improvement:** To maintain better the context I’ll include the first 100 chars and the last 100, usually the AI says the topic in the first statement.  
    
* The complexity of the topic increases the time of processing, questions about Marxism take more than 10 seconds.
* Apparently the only way to handle **timeouts in AWS Lambda** functions is manually in code, there’s no event handler.  
  \* <https://advancedweb.hu/how-to-handle-timeouts-in-lambda-functions/>
* I could call a **second lambda function** containing the communication with OpenAI multiple times with different amounts of data, each one with less data, and retrieve the first answer available, up to 3 requests (e.g.).  
  \* <https://www.sqlshack.com/calling-an-aws-lambda-function-from-another-lambda-function/>
  + The problem is that this would multiply the usage (1 + 0.7 + 0.4 = 2.1).
* I could use directly the **HTTPS** NodeJS **package** to make the request and **establish a timeout** for the request.  
  \* <https://stackoverflow.com/questions/6214902/how-to-set-a-timeout-on-a-http-request-in-node>
  + The *https* package uses the same options than *http*.  
      
    \* <https://nodejs.org/api/http.html#httprequestoptions-callback>

**\* Steps:** Basically a race… ****

* + Modify the code to use *https* node package using the timeout.
  + Retrieve a response if the response time exceeds 8 seconds.
  + Implement multiple requests with different parameters (less max\_tokens and smaller request).
    - One request with the entire…
    - I can split the AI answer in one block containing 100 chars, and another containing only 50 chars.
    - I could just send the current user request and not the last one.
* I can evaluate if the result of a request will be a list just sending the user message, but still the .
  + What happens if I include the “No list format” at the beginning of the text instead of the end.  
    \* Looks like instead of *\n\n* I could use *<|endoftext|>*.
* I noticed I can receive a fast answer 1s approx. when I make the request with very few tokens. Is there a way to predict the amount of time the request with more tokens will take?  
    
    
  *\* Certainly not using a quotient:*  
    
  
* I’m going to make **multiple simultaneous requests to OpenAI** without checking if the result if a list or not, I’ll just ensure “-” are replaced by comma “ , ” and that when a numbered list ends with a number, to remove it.
* **Two problems:**
  + GTP wasn’t completing the text as a conversation all the time. Sometimes it just continues completing the last messages.  
    **\* Solution:**
    - To use tags in order to differentiate the conversation participants, the user and the model.  
      
    - With this I’m able to change the personality of GTP.  
      
    - I also added a portion of text at the beginning of the prompt to specify the personality of the Model participant.  
      
  + GTP usually enters into a questioning loop, to every answer it responds with a question, or into boring conversation patterns.  
    **\* Solution:**
    - I added a random personality modifier.  
      
    - There’s a ½ probability of not permitting a question in the answer.  
        
      And from the two responses, I try to avoid the one that could be empty because of this change.  
      