Rodrigo Hurtado Molero

Chatbot with Server-side Logic

3/31/2019

Table of Contents

[Part I Serverless Architecture using Cloud Functions. 2](#_Toc4867562)

[Part II Watson Dialog Skill and Assistant 4](#_Toc4867563)

[Overview of the intents 4](#_Toc4867564)

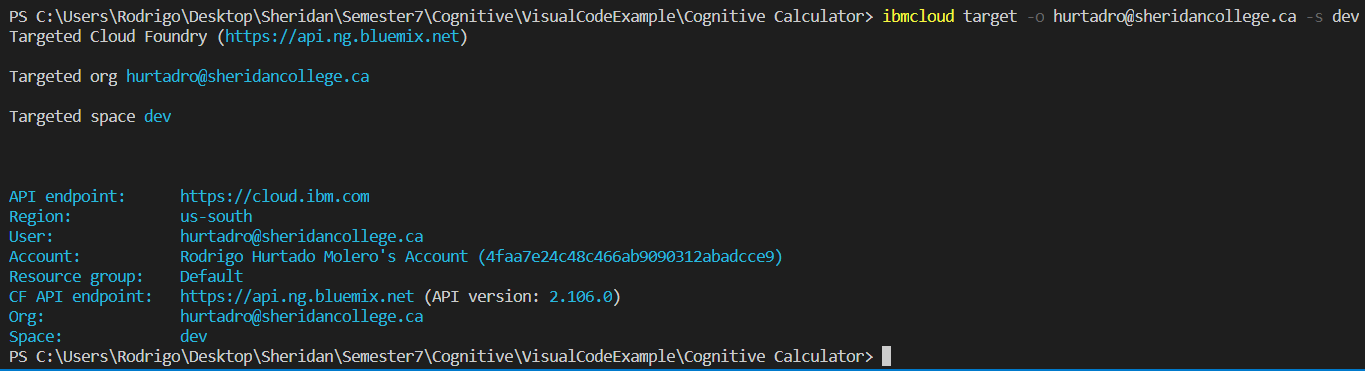
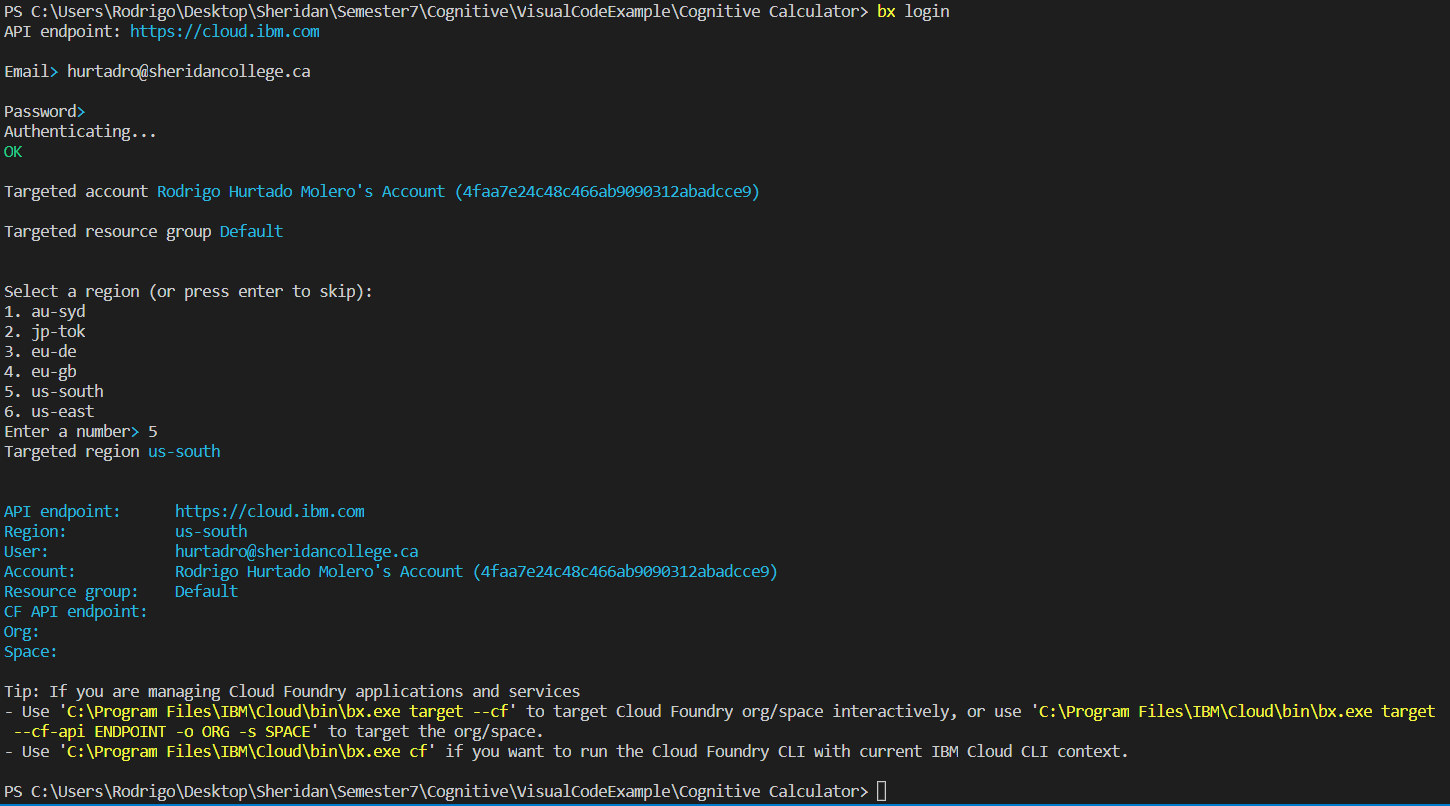
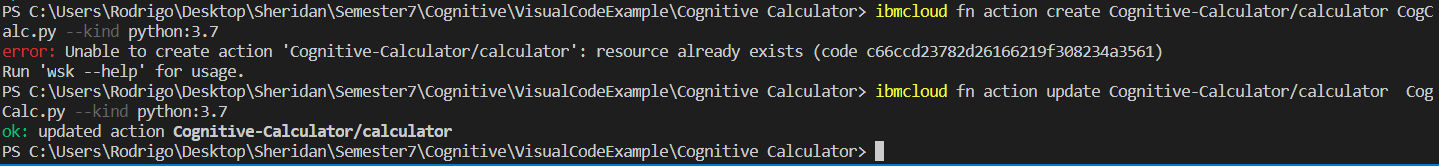
[Overview of entities 5](#_Toc4867565)

[Overview of dialog design 6](#_Toc4867566)

[Part III Slack Integration 6](#_Toc4867567)

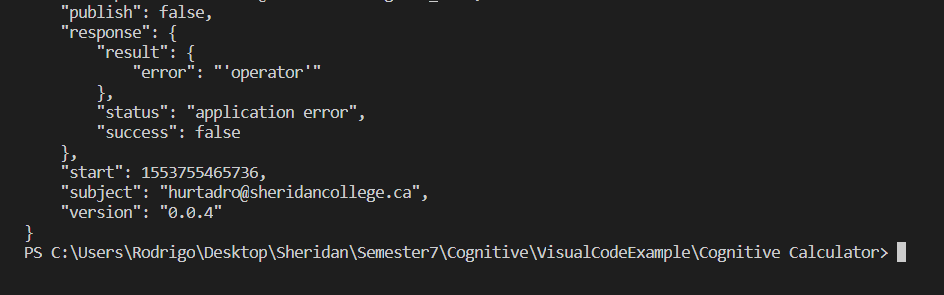
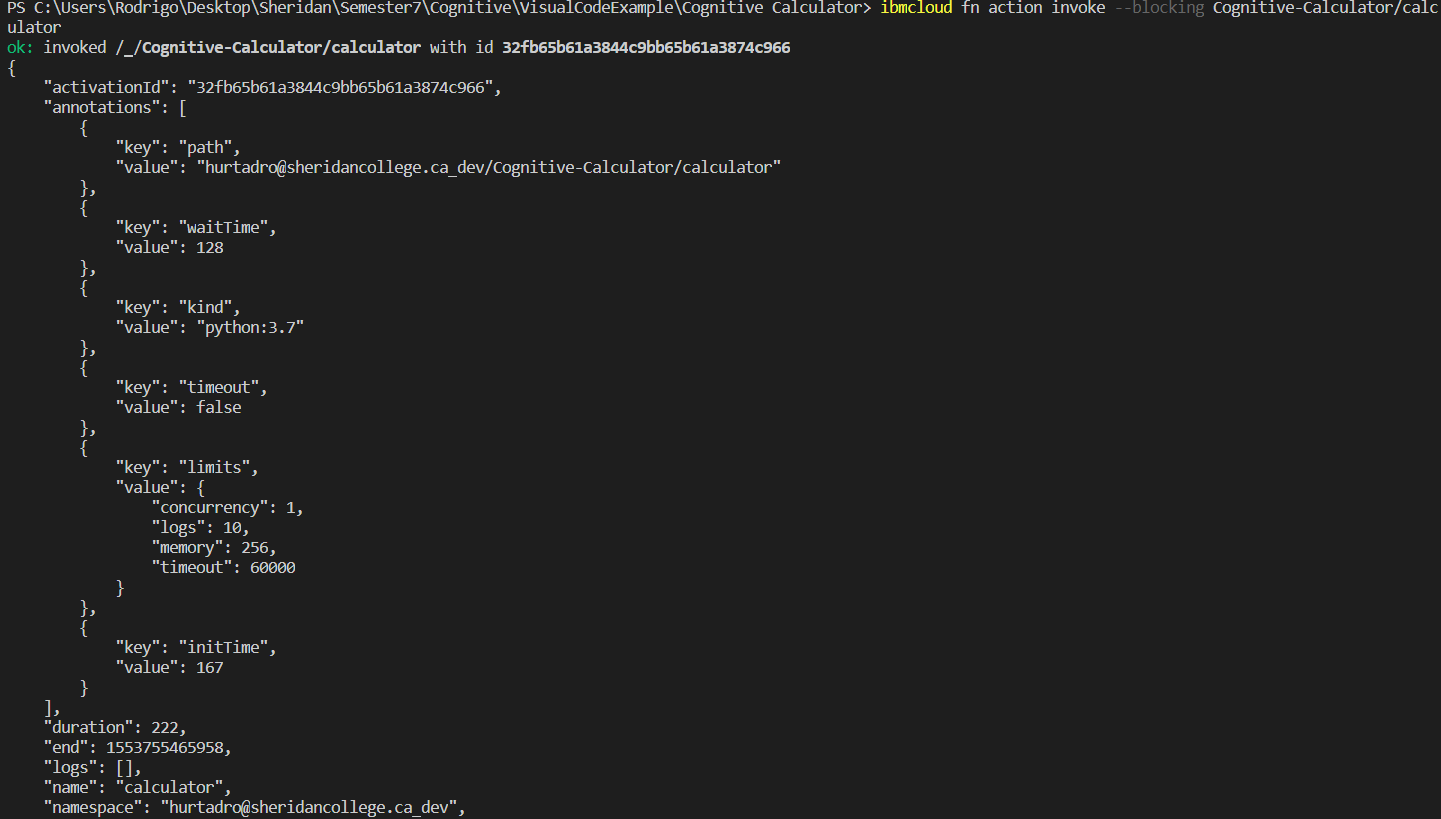
[Part IV: 9](#_Toc4867568)

# Part I Serverless Architecture using Cloud Functions.

Pictures of IBM Cloud commands used to create or update the cloud function including the login operation performed through the IBM Cloud CLI Plug-in for VS Code. 

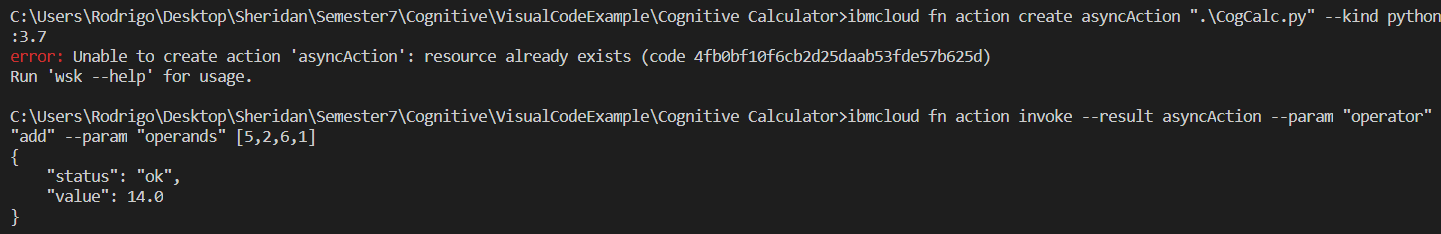
Creating did not work as it was already created. Computer restarted while working because of windows update, so I could not show that it said ok.

Example of invoking the cloud function in a blocking way:

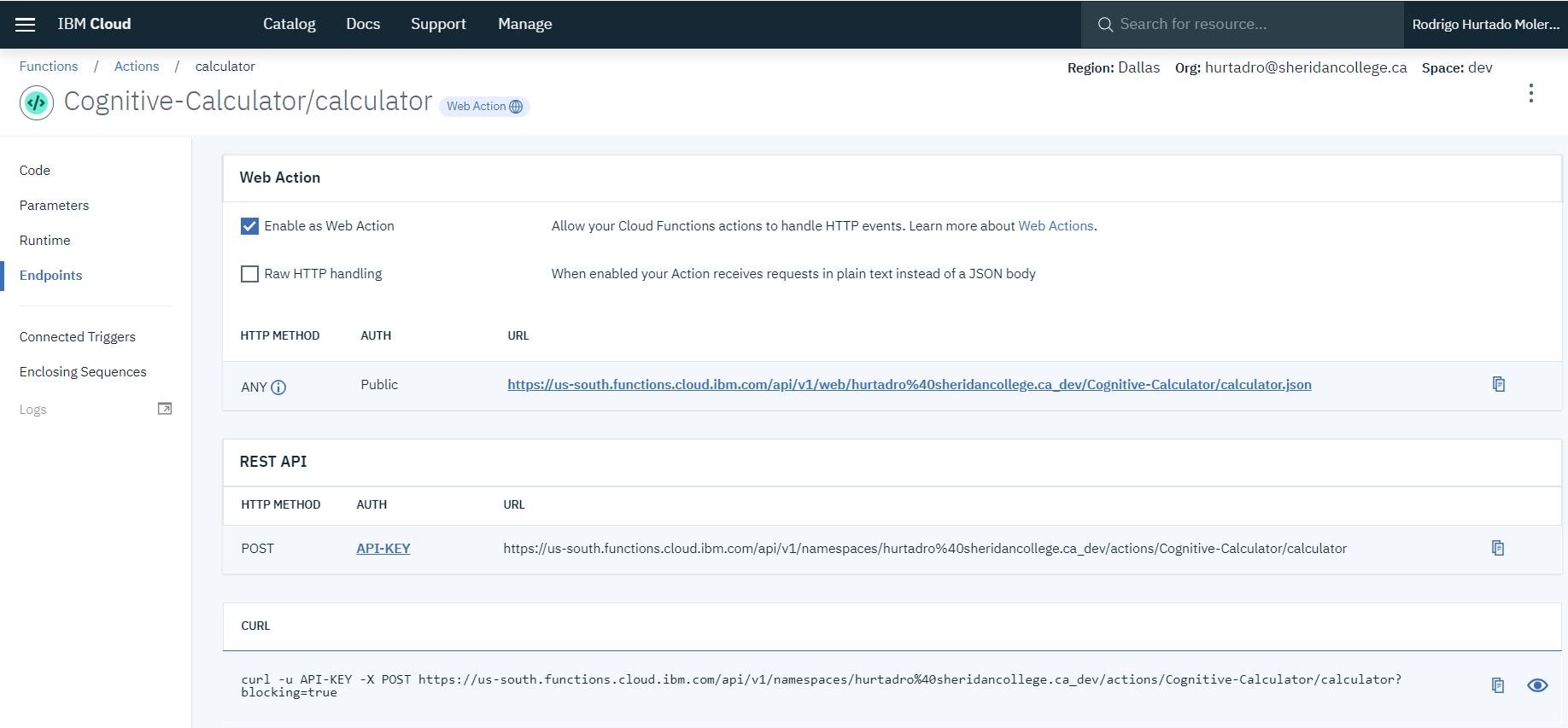


Example of invoking the cloud function in an asynchronous way

This gave me an error because I already created but I could not take a picture of It due to my computer having a window update at that time.



Description of the cloud function that was developed:



Description of the cloud function that was developed

The function that was developed takes in an operator and many operands to do simple math calculations such as addition, subtraction, multiplication, division, and average. Each function will take a list of numbers supplied as arguments and iterate through the list and do the functionality. For example, def add(operands) will take the list and add all the operands values in it. The function is then pushed to the cloud through CLI commands. The cloud function then is used as a back end for the conversation with the bot. The bot is design with Watson-Assistant and uses the cloud function to do the mathematical operations and displays the result back to the user. Therefore, the whole purpose of the cloud function is to provide a back-end logic for the math operations that Watson cannot perform.

# Part II Watson Dialog Skill and Assistant

## Overview of the intents

#add intent

The add intent purpose is to identify if the user is typing an addition operator into the bot. It has training data such as “+”, “add”, “addition”, “sum of”, etc.

#sub intent

The sub intent purpose is to identify if the user is typing a subtraction operator into the bot. It has training data such as “-”, “sub”, “subtraction”, etc.

#mul intent

The mul intent purpose is to identify if the user is typing a multiplication operator into the bot. It has training data such as “\*”, “x”, “multiplication”, “multiply”, etc.

#div intent

The div intent purpose is to identify if the user is typing a division operator into the bot. It has training data such as “/”, “divide”, “div”, etc.

#avg intent

The avg intent purpose is to identify if the user is typing an average operator into the bot. It has training data such as “avg”, “average of”, etc.

#memoryOperator

This intent purpose is to identify if the user wishes to provided operation with the memory value. The idea for this intent to is do operations with the memory. It provides examples such as “m1 +”, “m1 – “, and etc.

#result

This intent purpose is to identify if the user wishes to do calculations with the results from the previous calculation. It provides training data such as “average result”, “divide the result by”, “add result”, and etc.

#save

This intent purpose is to identify when the user wants to save the result into memory. It provides training examples such as “save result to memory”, “save”, “save to memory” and etc.

## Overview of entities

@result

This entity purpose is to identify when result variable is being called so that it can be used within operations. It allows for the bot to differ between the dialogue called result and the entity called result. This is needed as it helps when doing operations with memory and result being able tell when its being called for memory and result operations or when its just result operations being called.

@sys-number

This entity is used so that the chatbot can recognize the numbers that are being provided by the user when talking to the bot.

## Overview of dialog design

The dialog design starts with conversation start which gets the credentials for the cloud function. Then there is the welcome dialog which tells the user talking to the chatbot what the chatbot does. Then there are the operator’s nodes in the dialog which picks up if the user is doing either addition, subtraction, multiplication, division or average. Within each one there is a check to see if numbers were provided and if they were bot will ask to provide operands otherwise it will be sent to another dialog node which is called Cognitive Calculator call

Cognitive Calculator call node is the most important dialog of this whole chatbot. This dialog is the dialog that takes the operand and operator and send it to the cloud function to later get back the results based on the operator. It has checks to see which operator it is so that it will result in giving the right operator result.

Additionally, there is Saving number dialog which saves the result into memory so that the user can later call m1 to do operations with it. There is a memory1 with any operator which picks up on a user wanting to do an operation with memory and other operands and within it checks also for the result incase it’s a memory and previous result calculation. There is also a dialog that covers Result and any operator with the operand to do the calculations together.

There are a few limitations to the dialog. First it won’t do proper operands when dealing with result and m1 due to some dialog child nodes issues I came across. When doing operations with result or m1 the number is added to the last position in the list so say I want to do 3 + 2 + m1. The list will be [3,2, m1] but if I was to do m1 – 2 – 3, the list will be [2,3, m1] which will calculate it wrong. Also, the operator does not update properly due to the child nodes issue I was coming across and I had no idea how to fix it so I could not do an operator check within the m1 and result calculations. Additionally, when typing “2,3 and 4” by it self it will sometimes either prompt you with please provide operator and operand or it will do division operator. So, error handling does not properly work, though it worked sometimes but not all the time.

# Part III Slack Integration

Image below represent the welcome dialog

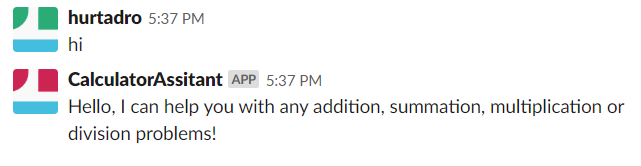


Image below represent the addition operation

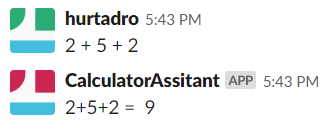


Image below represent the multiplication operation by providing only the operator first then asking for the operands

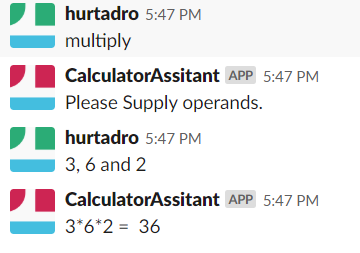


Image below shows division operator with 3 operands.

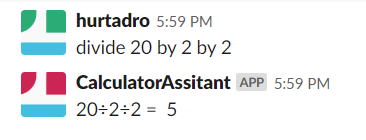


Image below shows average with no operands and after providing operands

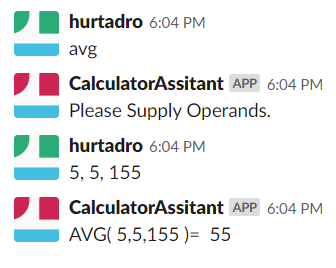


Image below shows saving memory 1.

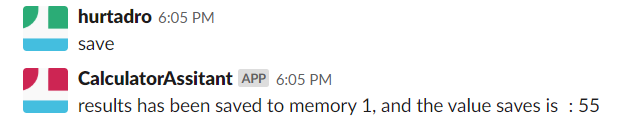


Image below shows m1 being added to other operands.

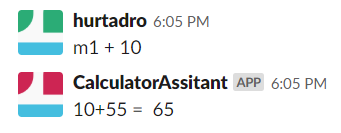


Image below show m1 being multiply by other operands.

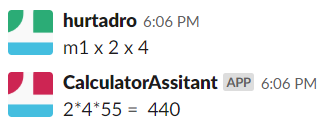


Image below show result being added with other operands.

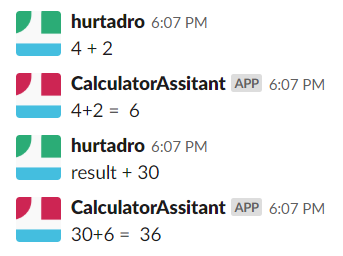
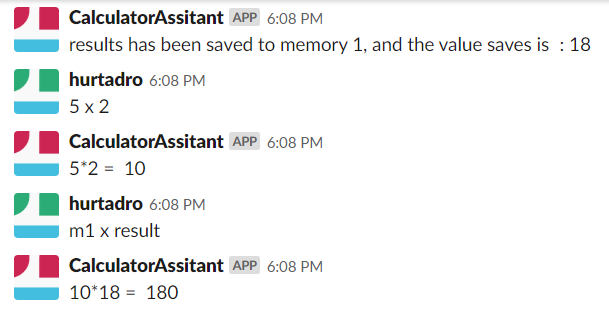
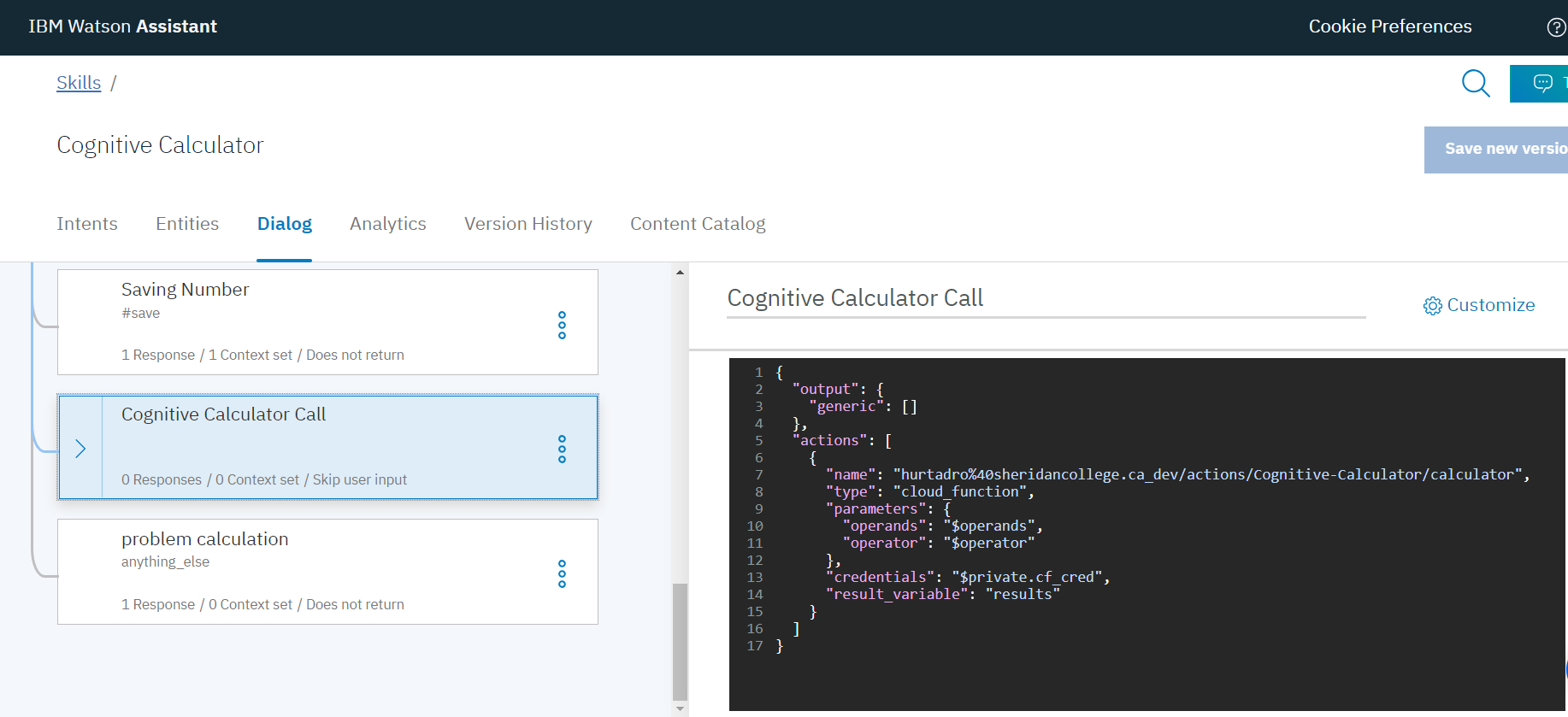


Image below show the results and m1 being multiplied together.



# Part IV:

1. One node is used to call the calculate cloud function in the dialog for all operation



Within this node there are 5 child nodes to provide a different result according to which operator was called. The parent node is the only node that calls the cloud function.