CHATBOT HACKATHON

OVERVIEW & PURPOSE

The goal of this hackathon is to build a conversational bot to interact with the user and achieve the desired outcome through the conversation. We shall take a two-pronged approach towards this task.

1. Use Amazon Alexa APIs to build a bot easily by defining the desired utterances and intents
2. See how a bot may be built from scratch and understand the underlying logic

UNDERSTANDING CHATBOTS

Understand the basic definitions of a chatbot and how Alexa Skills help in building them

- What is a Chatbot?

  https://towardsdatascience.com/architecture-overview-of-a-conversational-ai-chat-bot-4ef3dfefd5
- What is Alexa Skills Kit and how does it help with building a chatbot?

  The Alexa Skills Kit is a software development kit (SDK) that enables a developer to build skills, also called conversational applications, on the Amazon Alexa artificial intelligence assistant

GETTING READY

- Create an **Amazon Developer Account:**
  - To build any type of skill, you need an Amazon developer account. Go to the developer console at https://developer.amazon.com/. You can use an existing Amazon account to sign in, or you can create a new Amazon account. Follow the prompts to enter your registration information and give consent to the Amazon Developer Services Agreement. Once you have done this, you can build skills for Alexa using this account.

- Learn the **Nuances of dialog management in Alexa :**
  - This is critical in building a good, interactive, and intelligent chatbot. Efforts in understanding this will contribute to building a good chatbot. A clear understanding of the different aspects of Alexa Dialog's management will help to crack the hackathon easily.

TO UNDERSTAND ABOUT AMAZON ALEXA

Go through the below links:

- In this link, you will understand how to build Alexa.
- Click here to understand how utterances, Intents, and slots are used in Alexa.

CHATBOT ARCHITECTURE

The chatbot is made to be provided as a pluggable service, and configurable through configuration files, in addition to modifying code.

CONCEPTS

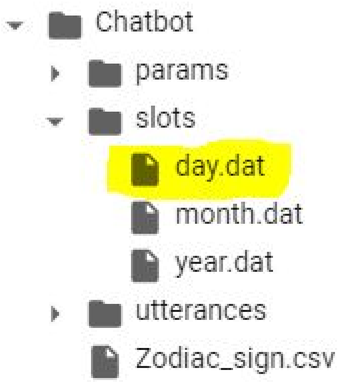
Intent: This is a particular skill of a chatbot. It could be a get Zodiac Sign or Movie Suggest, etc. Each intent has a few **parameters**, which are basically information required to be elicited by the user for the chatbot to complete the particular intent. Example: birth date of the user, language preferences, etc.

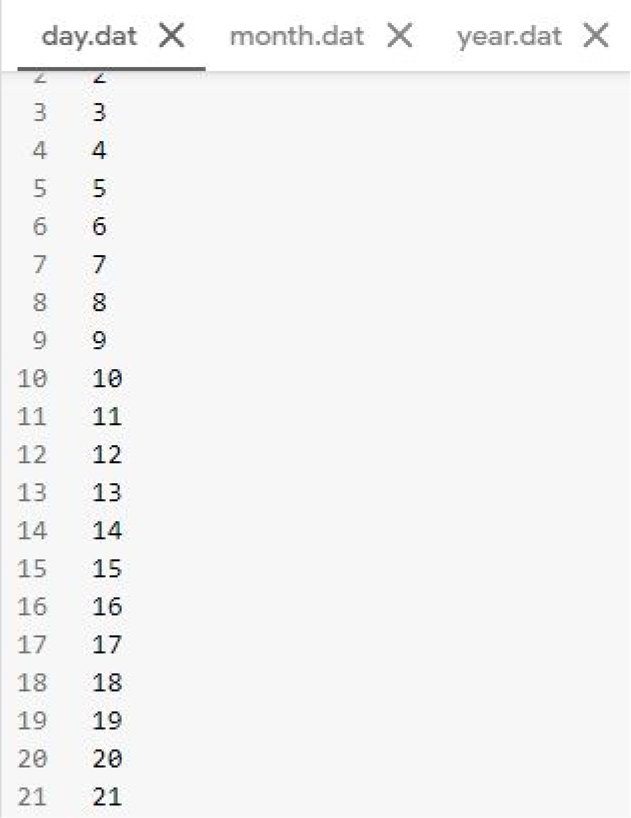
Actions: Each intent also had action, that is a function call after all the parameters have been fulfilled.

CONFIGURATION FILES

Slots Folder:

Here each file contains example slots.

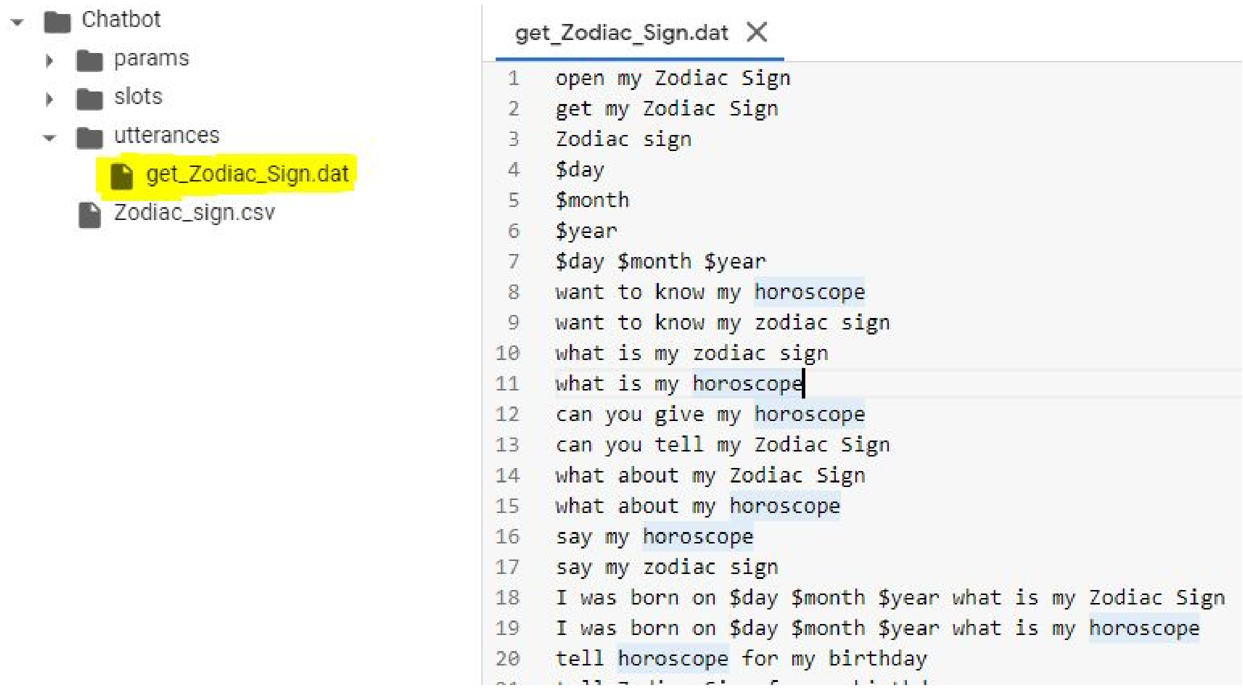




```
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
```

Utterances Folder:

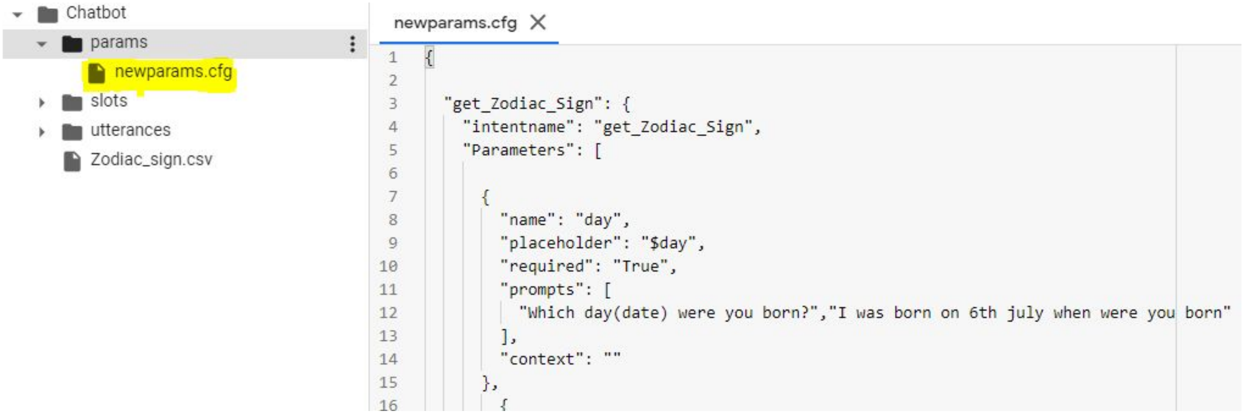
Here each file contains example utterances for each intent. Note the placeholders starting with $for entity masking (as done by get_attributes())



```
open my Zodiac Sign
get my Zodiac Sign
Zodiac sign
$day
$month
$year
$day $month $year
want to know my horoscope
want to know my zodiac sign
what is my zodiac sign
what is my horoscope
can you give my horoscope
can you tell my Zodiac Sign
what about my Zodiac Sign
what about my horoscope
say my horoscope
say my zodiac sign
I was born on $day $month $year what is my Zodiac Sign
I was born on $day $month $year what is my horoscope
tell horoscope for my birthday
```

params/newparams.cfg :

This file contains the JSON which holds the parameters for each intent, and its corresponding values, such as placeholder, required, etc.

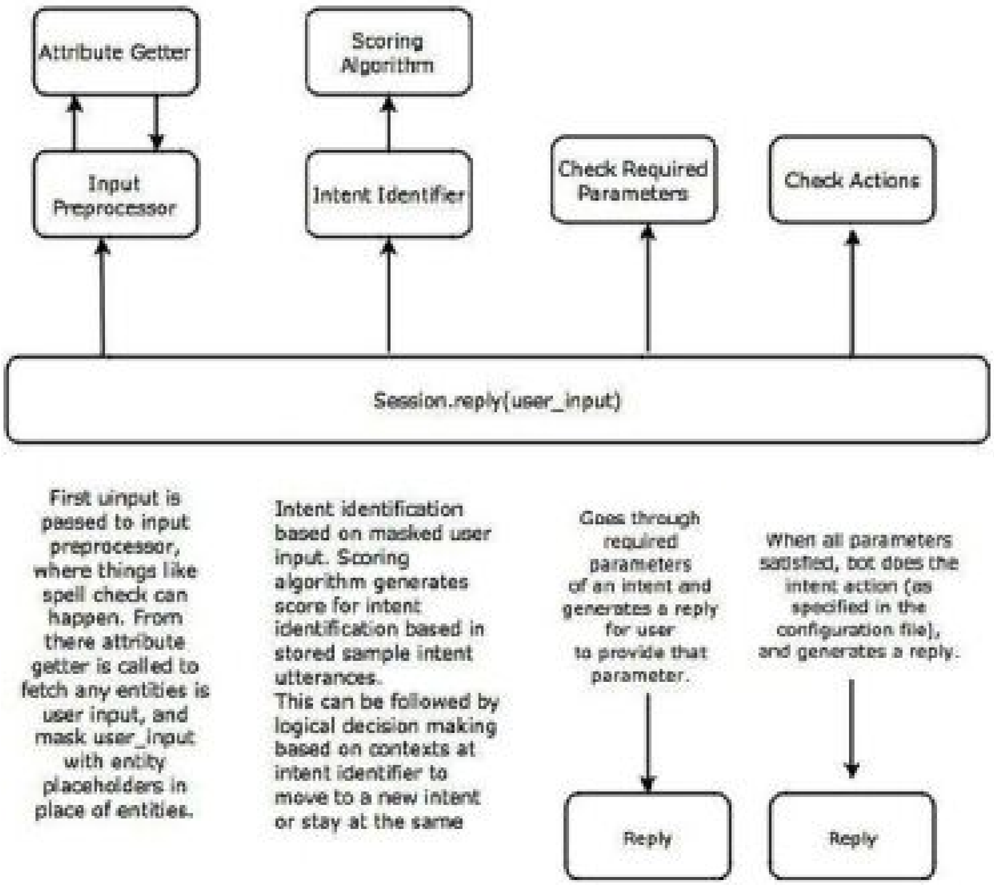


```
{

  "get_Zodiac_Sign": {
    "intentname": "get_Zodiac_Sign",
    "Parameters": [

      {
        "name": "day",
        "placeholder": "$day",
        "required": "True",
        "prompts": [
          "Which day(date) were you born?","I was born on 6th july when were you born"
        ],
        "context": ""
      },
      {
```

Class Objects

- **Contexts** (context.py) - Here contexts are some flags that we switched on with each of the parameters or these can be used in conditional if-else statements. Below are the functions:
    - activate_context( ) and
    - deactivate_context( )
- Classes which are inheriting the properties of Context
    - FirstGreeting( ),
    - IntentComplete( )
- **Intents** (intent.py) - Every Intent has its name and particular action and parameters. Parameter class is used to hold the attributes of intent from the JSON file.

CONVERSATION FLOW



First uinput is passed to input preprocessor, where things like spell check can happen. From there attribute getter is called to fetch any entities is user input, and mask user_input with entity placeholders in place of entities.

Intent identification based on masked user input. Scoring algorithm generates score for intent identification based in stored sample intent utterances.
This can be followed by logical decision making based on contexts at intent identifier to move to a new intent or stay at the same

Goes through required parameters of an intent and generates a reply for user to provide that parameter.

When all parameters satisfied, bot does the intent action (as specified in the configuration file), and generates a reply.