Semantic web engineering assignment 2 student ID: 51555097

Task 1 and 2

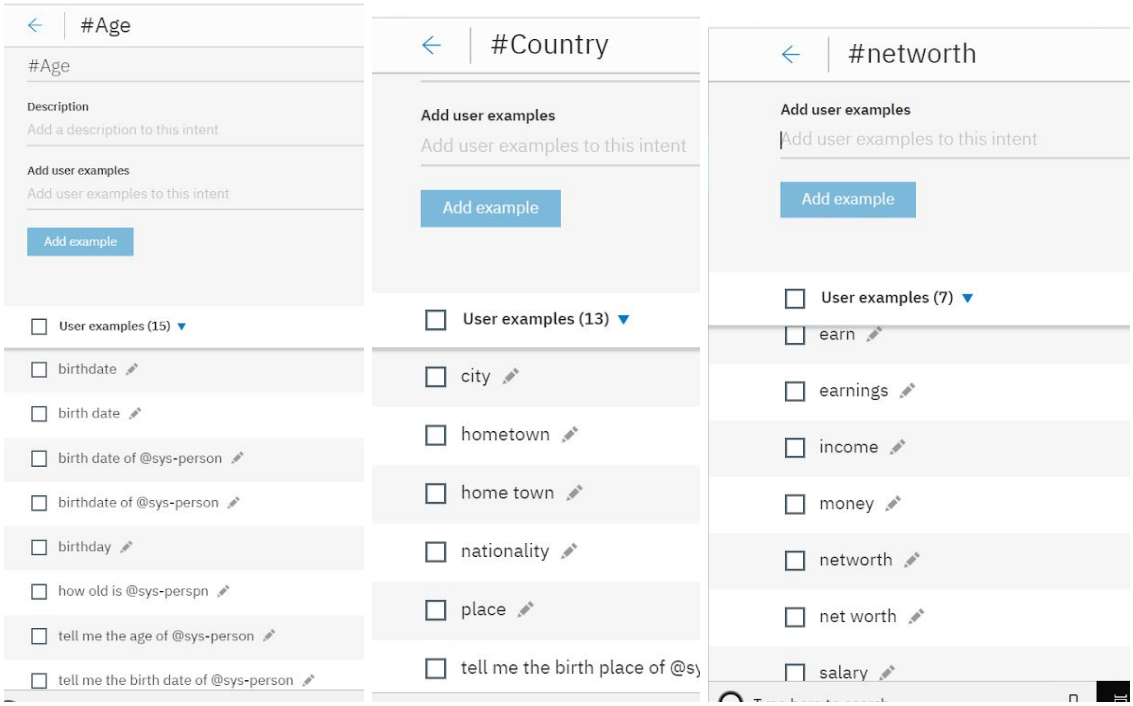
Dialog systems have become a very important part of the web because they provide users with useful information. These systems give users the feeling that they're talking to a real human because they contain a conversation flow. This aids human convenience because they can receive instant and reliable results to their queries instead of having to actively search through the web to get their answers.

My dialog system will be able to answer questions about Presidents, Politicians and Billionaires. The system will be able to answer questions on attributes:

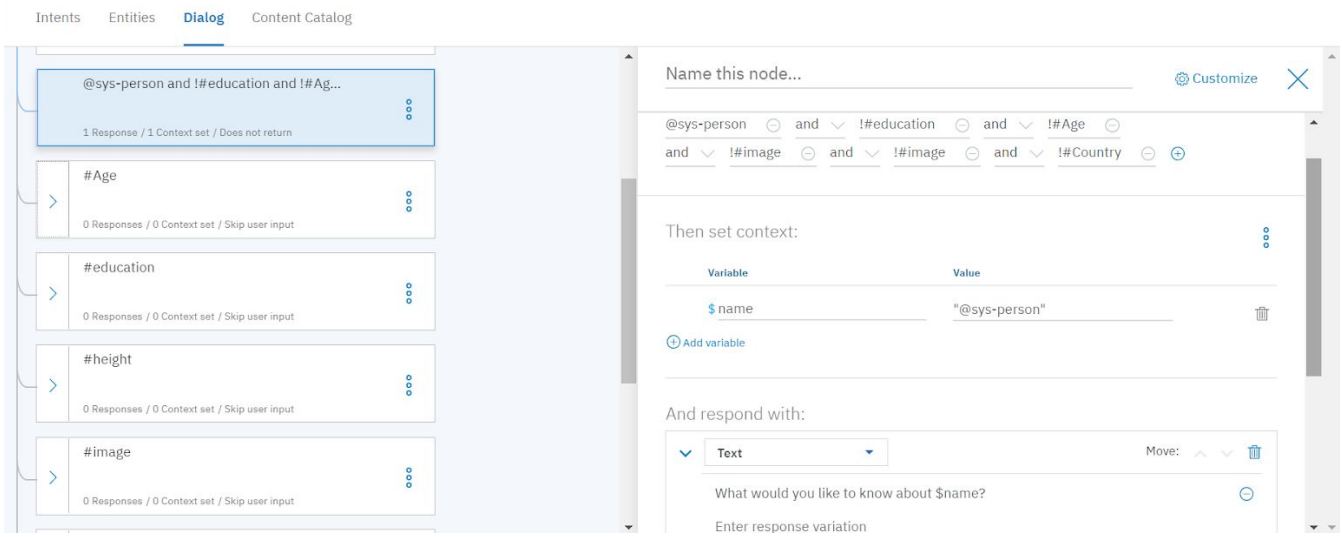
1. Birth date
2. Birth place
3. Spouse/partner
4. Graduation details
5. Description
6. Net worth
7. Party (political)
8. Children

The system will be able to answer questions such as "Who is Hillary Clinton?", "What's Mark zuckerberg's Net worth?", "Who is Xi Jinping married to?" Etc. The first very thing i did was to research on which linked open data and ontologies i wanted to use and what i wanted my dialogue to be about, Based on that, i designed my dialogue. Dbpedia's ontology is used to query results. Yago Subclasses; yago:President110467179, yago:Politician110450303, and yago:Billionaire110529684 have been used so that the application is able to query about anyone who is a politician, president or billionaire.

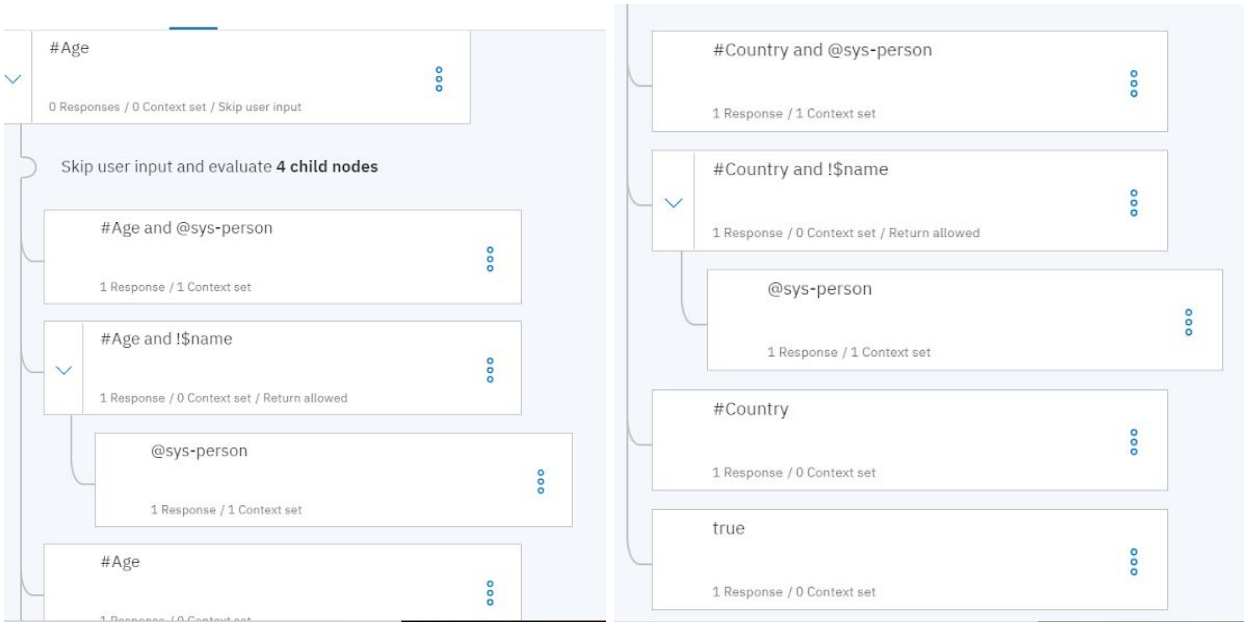
Below are a few screenshots of how the dialogue looks on IBM watson assistant.
Point to note: This was my initial dialogue and a lot of new indents have been added and removed since then. A lot of trial and error had been done in order to make the dialogue work the way i wanted it to work. The responses have also been replaced by SPARQL query placeholders now. The dialogue was initially designed to only answer questions about politicians but now it can answer questions about politicians, presidents and billionaires.



As you can see in the picture above, i have used a lot of keywords for each indent so that its easy to detect which indent the user is talking about in the dialogue.

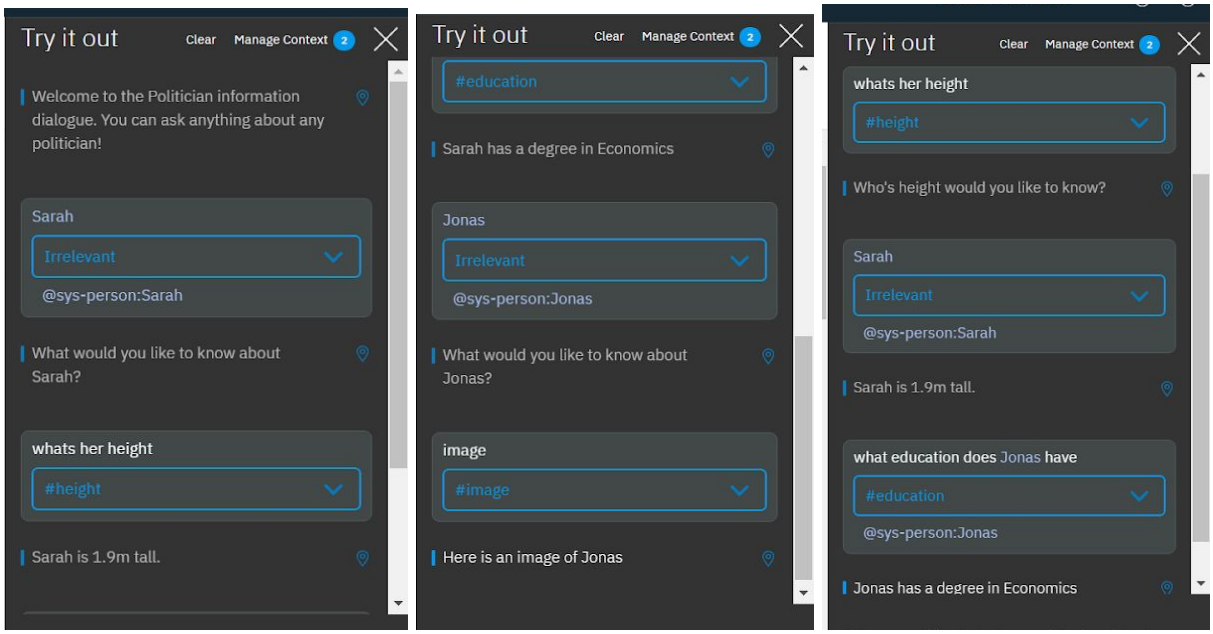


The indents used are age, education, height etc. Each indent has a variety of conditions as you can see in the picture below:

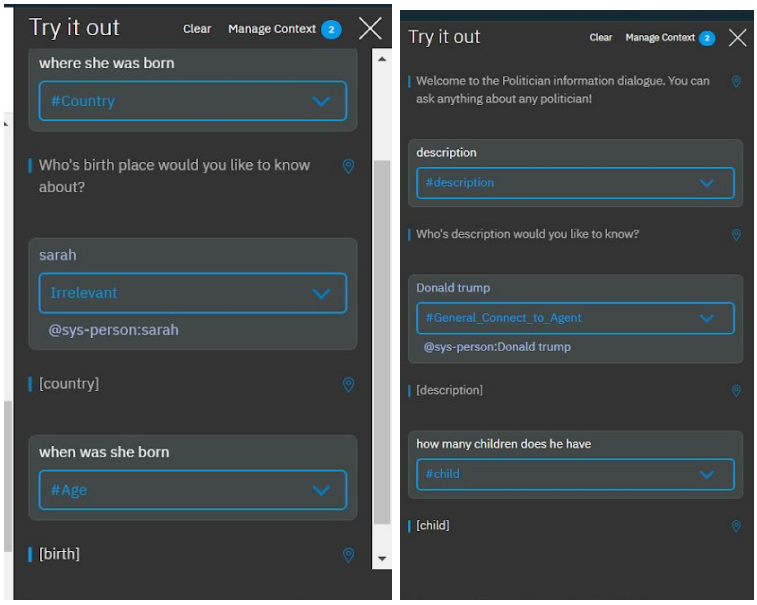


Based on the condition, responses are fired. For example: first condition **#age and @sys-person** means that if the user asks a question about a person's age and the name of the person is given in the question, then set the @sys-person entity to context variable $name and output a response. Another example, the condition **#Age and !#name** is fired when a user asks a question about a person's age but the name of the person isn't provided. So, in this scenario, the dialogue asks the user 'Who's age would you like to know about?' and after the user types in a name, the dialogue then sets that @sys-person into $name and outputs "$name is 20 years old". Third condition **#Age** is fired when $name variable already exists in the context. Similarly, the rest of the conditions are fired based on how much information the user provides in their input question.

The only context variable used throughout the dialogue is $name and the only entity used is @sys-person. This is because the way my dialogue system is designed, these two are the only ones that are required.



Now the dialogue looks the one below: It no longer contains dummy response values. It now contains SPARQL place holders that are checked by the node.js application in order to give responses to the user based on results obtained back from SPARQL queries.



I have included a JSON file in the submission folder. If you would like to view the dialogue on IBM watson assistant, the skill name is 'semantic_web_assignment. Username: rutushaah@gmail.com and password is Rutu_shah12345 (or Rutu_shah123). Here is a link: https://assistant-us-south.watsonplatform.net/eu-gb/5cf3273e-2973-4681-9fd4-26952dda7b82/skills

Task 3: To design the application i used node.js. To set up the IBM assistant connection, i followed steps provided on the link here: https://console.bluemix.net/docs/services/conversation/develop-app.html#implementing-app-actions

In places where the dialogue has to respond, i added [birth] for example and then checked this condition in the .js file after which a query is executed to get results of the person's birth date from DBpedia.

```
//using UNION to query for presidents, politicians or billionaires
if (response.output.generic.length != 0) {
  if (response.output.generic[0].text == "[birth]") {
    var query = `PREFIX dbp: <http://dbpedia.org/resource/> PREFIX foaf: <http://xmlns.com/foaf/0.1/> PREFIX yago: <http://dbpedia.org/
    SELECT ?birth where {{ ?person a foaf:Person . ?person foaf:name ?name. ?person rdf:type yago:President110467179.
    ?person dbo:birthDate ?birth.
    FILTER regex(?name, "` + response.context.name + `", "i" ) .}
    UNION { ?person a foaf:Person . ?person foaf:name ?name.
    ?person rdf:type yago:Politician110450303. ?person dbo:birthDate ?birth.
    FILTER regex(?name, "` + response.context.name + `", "i" ) .}
    UNION { ?person a foaf:Person . ?person foaf:name ?name.
    ?person rdf:type yago:Billionaire110529684. ?person dbo:birthDate ?birth.
    FILTER regex(?name, "` + response.context.name + `", "i" ) .}}  `;
    dps.client()
    .query(query)
    .asJson()
    .then(function(r) {
      if (r.results.bindings.length != 0) {
        console.log(response.context.name + "'s birth date is " + r.results.bindings[0].birth.value + ".");
      } else {
        console.log("Sorry! I can't find the Birth date of " + response.context.name + " in the data source. Please ask me something e

      }
```

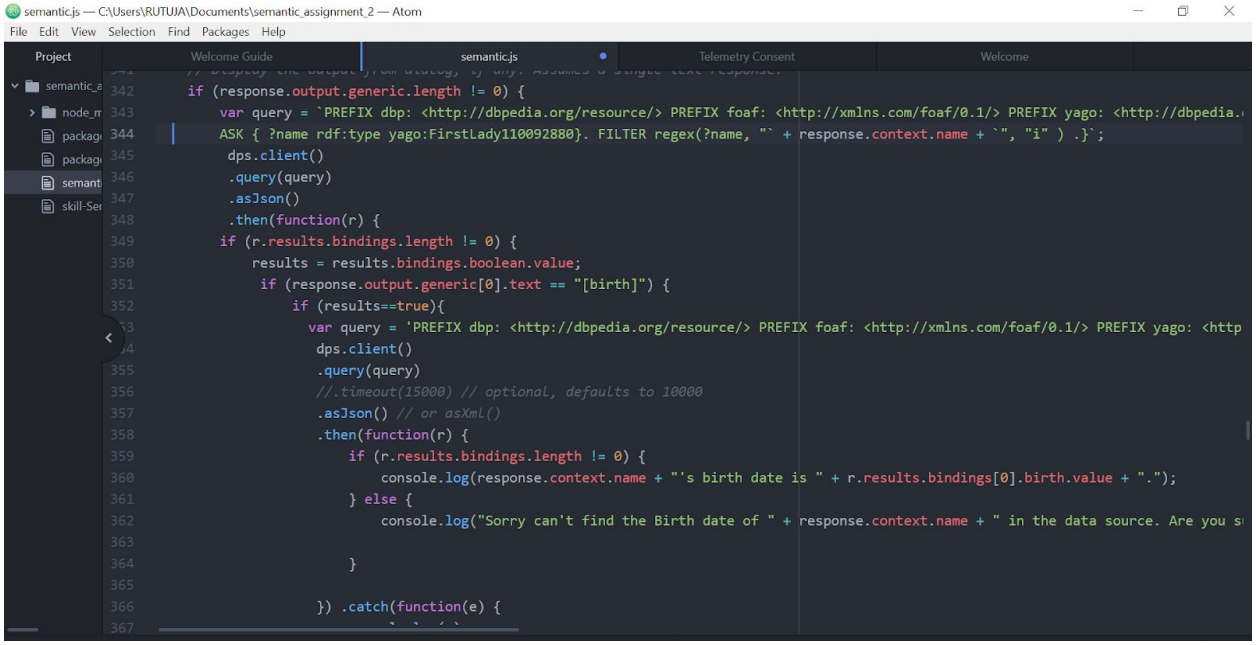
As you can see in the editor above, when [birth] is received as a response from the dialogue, the query above is executed. The query above is very long because i have used UNION for capturing different query patterns so that the dialogue can fetch results for presidents, politicians or billionaires ontology classes. So if a user asks a question regarding the birth date of Hillary Clinton, then the query pattern for yago:Politician110450303 will be matched.

Context variable $name from the dialogue is used here so that the query is specific. Using this approach makes the queries dynamic because you can query about any one as long as they are a part of the politician, president or billionaire ontology class. The name will be simply taken from the dialogue and plugged into the query to fetch results about that specific person.

Results from the query have been concatenated with strings to structure the output response and additionally, 'else' statements are added for cases where the query was not able to fetch results. This could be because the person queried for is not a part of the politician, president or billionaire ontology class, it could also be because that person does not have the named attributed in DBpedia. For example: Mark Zuckerberg does not have **dbo:party** on his DBpedia page and hence the query can't fetch it.

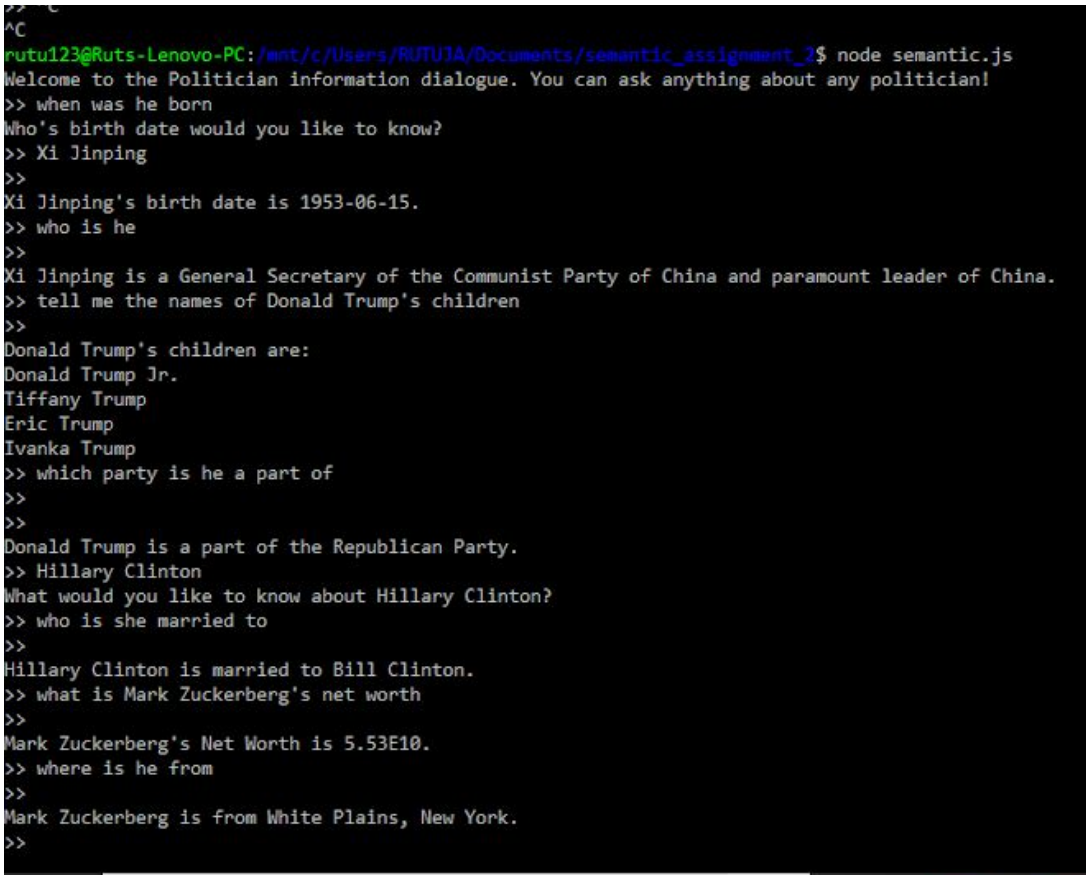
I tried doing an approach involving an ASK query, So ideally, what i was trying to do is to first get the person's name from the context variable and do an ASK query on it to first check if the

person is a politician, president, billionaire or not. Only if the person belongs to one of these groups would then the response condition such as [birth], [country[be checked but otherwise, an output response would say “Sorry, the person who are asking for is not a politician, president or billionaire. Please ask about a person who is a politician, president or billionaire.“ Below is a screenshot to prove that i tried to do this but i was unable to do it because i got errors that i couldn’t fix after loads of trying: The below didn't work so i had to remove all of it. But i wanted to show you that i tried.



So, each indent on the dialogue has a response place holder such as [birth] for example and each of these place holders are checked in the node.js application and based on that, the queries are fired to give results back to users. In places where there are more than one results for example, Donald trump has 4 children, a **for loop** is used to fetch every result. Screenshot below show how these for loops are used:

I did not get time to make this dialogue work on a browser. So this is all i have done, i'm sorry. When the .js file is run on the console, the dialogue works on the console itself. There are many glitches on the console. The dialogue works fine when no queries are involved but when it has to query something, you have to keep pressing ENTER in order to get results. The results are accurate but i tried fixing the glitch but there is nothing i can do about it. Below is a screenshot of how the dialogue looks on the console:



'>>' is what i have typed on the console. There are blank ones because i kept pressing enter to get responses from the dialogue. Other than the glitches, you can see that the dialogue gives out correct responses. Here i have used a mixture of billionaires, politicians and presidents. Mark Zuckerberg is a billionaire, Hillary clinton and Xi Jinping are politicians (and also presidents).. Donald trump is a billionaire (he is not a part of the president or politician class on DBpedia). Thanks to UNION, my queries were able to fetch results for all of these people belonging to different classes.

Because of this glitch issue, i had to remove the '#Anything_else" indent from IBM assistant because every time i pressed enter, the result of the query along with "i don't understand, can you rephrase?' was outputted. This is because pressing enter triggers the '#anything_else' indent because its an empty input. and hence to avoid more disturbances on the dialogue, i had

to remove this indent. So now if you type something out of context, the dialogue won't respond with anything. I know this is not a good practice but to avoid from disturbances, i had to remove this indent.

How to run the application:
Once you have node.js installed, all you have to do is type **node semantic.js** on your terminal (once you're in the same folder) to get the dialogue working on it. The submission folder already contains all the node libraries that are needed for the application to work. To view code, please open it from ATOM because backticks have been used and atom supports this whereas notepad++ does not.
If there are tunnelling errors, run file on linux/unix.

Improvements that could be made on current system::

1. To bring the application on the browser instead of using it on the console
2. To fix the glitching issue and bring back the anything_else intend so that irreverent user inputs have a response from the dialogue.
3. To have an ASK query that first checks whether the person being queried is indeed a politician, president or billionaire. Only if the ASK query returns TRUE should other conditions such as [birth], [country] etc be checked. Because there is no point in executing a query about for example 'Birth date of Bob marley' because Bob marley is a musician. So, in this case, the system would get a FALSE from the ASK query and user would be notified that the person they're asking about isn't a politician, president or billionaire and that they should only ask questions about politicians, presidents or billionaires. (i did try this as shown in the report but i was unable to do it and given time constraints i could not spend more time on it).
4. To add more intends example: signature, parents, image etc. I only provided 8 because it is all just replication and i didn't want the code to be too long.

My initial plan was to somehow make the entire dialogue dynamic, Meaning, to design it in such a way that it is able to answer questions about any attributes of a specific class. No fixed number of indents would need to be defined. For example, if i was to make the dialogue about universities, then the dialogue would be able to answer any questions about any universities around the world. Indents would not be hard coded as done in the current system but instead indents would be captured as the user asks something on the dialogue and based on that input sentence, keywords would be extracted and queries would be run on these.

For example: 'ranking' would not be an intend on the dialogue but when user asks 'what ranking does university of aberdeen have', the keywords 'ranking' and 'university of aberdeen' would be extracted and put into variables which would then be used to query results. For this to work, a lot of natural language processing ideas would be needed to carefully design a system that can answer any question about any university without having a fixed set of indents that my system currently has. This is my own idea and when i have time in the future, i will build such a system.

Lessons learnt:

I was unable to bring the dialogue results on a browser for better convenience but this is only because i ran short of time. This semester has been very busy so i've only allocated a few days to each task. What i have learnt from not being able to finish this project is to fully complete tasks given int practicals sessions and to better plan my schedule.