Assignment 2

# N Mohammed Sohaib

# 200611

## Solution Documentation :

**Methodology:**

1. **Understanding the Problem**: The first step for us to start is to thoroughly understand the problem statement and the requirements. In the given case, the task for us is to design a QA system that interprets natural language queries and generates structured data responses based on predefined datasets.
2. **Analyzing Test Cases**: Reviewing the provided test cases helps in understanding the types of queries the system needs to handle and the expected output format. For example,

"Which ships have a grid with more than 50 GB RAM?"\*

1. Identifing Key Components:

- Retrieve ship names.

- Condition on it is Grid RAM > 50 GB.

2. Maping the Components to Data Structure given to us :

- Table: grid\_details

- \*Column to Select : ship\_name

- Condition: grid\_ram > 50

3. Construct an output based on the above conditions :

- select\_column: ship\_name

- from\_table: grid\_details

- where: [{'column': 'grid\_ram', 'relation': '>', 'value': 50}]

1. **Buildinging a Solution**: Like this the above test cases should be handled wisly to give the desired output and again the solution involves processing natural language queries to extract relevant information such as the select\_column, from\_table, and conditions for the where clause. This information is then used to construct a structured data response.

**System Architecture:**

1. **Input Processing**: The system first tokenizes the input query using NLTK's word\_tokenize function. It then removes stopwords and non-alphanumeric tokens and performs stemming to normalize the words.
2. **Query Parsing**: Based on the keywords and context within the query, the system determines the select\_column, from\_table, and conditions for the where clause.
3. **Output Generation**: Using the parsed information, the system constructs a structured data response in the form of a Python dictionary.
4. **Test Case Execution**: The system iterates through the provided test cases, processes each query, and compares the generated output with the expected output.

**Decision-Making Process:**

1. **Selecting Relevant Information**: The system identifies keywords and context within the query to determine the relevant information such as the select\_column and from\_table.
2. **Handling Ambiguities**: Ambiguous queries are handled by defaulting to certain assumptions. For example, if the query does not specify a column explicitly, the system defaults to 'ship\_name' for ship-related queries and 'grid\_name' for grid-related queries.

**Remarks;**

### My python code shows error to some other test cases, it’s totally my fault for not able to complete on the given certain time, I worked very diligently in the initial 2 days and final day, all day and night, as I have asked for the extension in deadline as well which was given I am very thankful to the HR, unfortunately my side I had to face various difficulties due to various misfortunes recently, I really want to work on this company but I could not able to show my worth now, I am sorry Its totally my fault ,Thanks .