SHARATH SREENIVASAN
STUDENT ID - 200109355

SOCIAL COMPUTING CSC 591 P2 (A) REPORT

The implementation approach I followed to solving the problem is as described below.

The project was implemented with the help of tweepy, a python library for accessing the twitter API. After obtaining the Access Token, Access Secret, Consumer Key and the Consumer Secret for the given application, the authorization for the access is acquired. The StreamListener class filters all tweets with the hashtag 'P2CSC555F15' for further processing.

CHECK – IN

The user's location is pre determined from a set of given locations. First, if the another person checks in at the same position as that of the user, then the user responds to the other person with a check-in of a particular format.

For example, the user's location was set as hunt
One of the check-in tweets that was the input –
I checked in at #hunt #id_cpancha_5 #P2CSC555F15

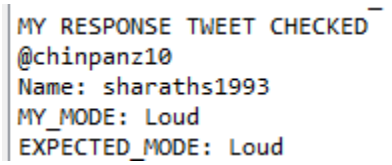


Fig 1 – Output response to check-in from neighbours

The word 'checked' was searched for in the tweet. The string between the first hashtag and the second hashtag was checked. If the string matched the user's location, the response tweet was constructed accordingly by getting the screen name from the json object of the tweet. The constructed tweet is then updated as a status for the user. The checked-in tweets are appended onto the file 'checkedtweets.txt'.

MAKE – CALLS

To check if the user can request an incoming call, the user responds with a tweet.

For example, one of the call requests is as follows –
CALL #id_yjmehta_15 #P2CSC555F15

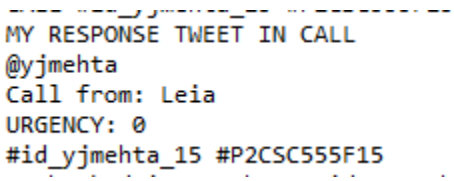


Fig 2 – Output response to call

The word CALL was searched for in the tweet. The response id is the substring between the two hashtags of the tweet. The Call from field is filled randomly from a pre defined list of callers. The urgency

is also chosen randomly. The call tweets are written into the 'calltweets.txt' file. The tweet response is then constructed and the status of the user is updated.

FEEDBACK FOR ACTION

For each call received, the action taken has a feedback from the neighbours. For this, the utility function has to be used and the action n should be determined based on the inputs to the utility function. Parameters such as the noise, location should be taken into consideration. Currently, the program chooses a random value from the list and the status is updated accordingly. The tweets are written into the 'actiontweets.txt' file. A snapshot of the output as the feedback for the action is

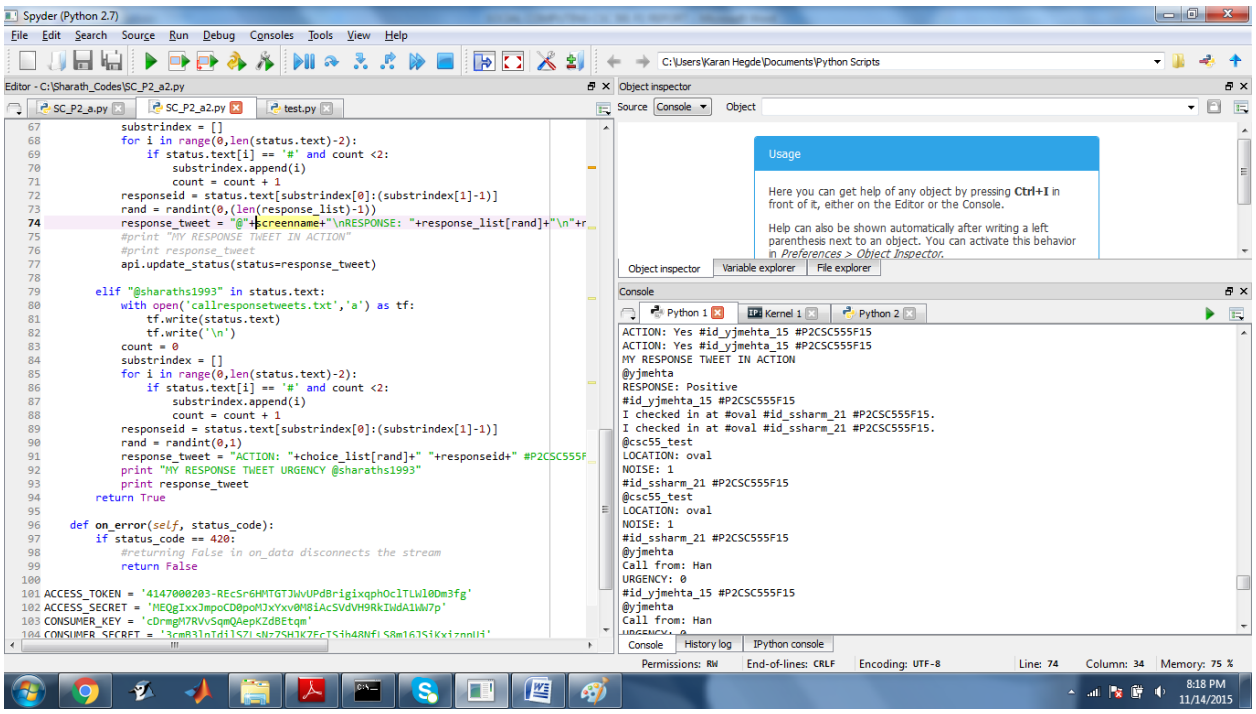
```
ACTION: Yes #id_yjmehta_15 #P2CSC555F15
MY RESPONSE TWEET IN ACTION
@yjmehta
RESPONSE: Positive
#id_yjmehta_15 #P2CSC555F15
```

Fig 3 –Neighbor feedback for action taken

CALL RESPONSE

For each call received, the user should indicate the action was taken. In this case, the utility function comes into use. Depending on the various parameters such as the strength of the relation, type of relation (family, friend, colleague, stranger) the corresponding action should be performed. In this case, the values are chosen at random. The filtered tweets were further selected as the call statements are directed at the screen name of the user. Accordingly, the response is built by obtaining the values from the call and the status is updated.

A few screenshots of the output are as shown below –



SHARATH SREENIVASAN
STUDENT ID - 200109355