Custom Code

Kevin Neilson, Marissa Le Coz, and Evan Honnold

November 13, 2016

We wrote several hundred lines of javascript code for this project. We copied Arman's "Conversation" project and edited the **app.js** file to handle conversation responses and make API calls to our Retrieve & Rank service. To make our code easier to review, we've provided this listing of just the code that we wrote ourselves.

To see our complete code base, please visit our Github repo: https://github.com/themacexper

1 Main Function

This function is called before the Conversation service displays a response to the user. If appropriate, it extracts the intents (symptoms), makes a call to Retrieve & Rank, and updates the response to include the list of relevant diseases.

```
/**
 * Updates the response text using the intent confidence
 * @param  {Object} input The request to the Conversation service
 * @param  {Object} response The response from the Conversation service
 * @return {Object}          The response with the updated message
 */
function updateMessage(res, input, data) {

   // extract symptoms
   var newSymptoms = extractSymptomList(data);
   permanentSymptomList = permanentSymptomList.concat(newSymptoms);
   console.log("Current symptoms list: " + permanentSymptomList);

   /** use one of these two lines depending on which way
   of keeping track of symptoms we're using**/
   var symptomList = permanentSymptomList;
   //var symptomList = data.context.symptoms;

   // if there's a "no [more symptoms]" intent, then gather up the symptoms
   // and make a call to the Retrieve and Rank API
   if (hasIntent(data, "no")){
      symptomList = removeDuplicates(symptomList);
```

```
23
24      // generate a retrieve & rank query by combining all the symptoms:
25      //var collectionName = "Neurological";
26      //var collectionName = "example_collection";
27      var collectionName = "neuro_collection";
28      var rankerID = "54922ax21-rank-20";
29      var query = "";
30      for (var i = 0; i < symptomList.length; i++){
31         query += symptomList[i];
32         if (i < symptomList.length - 1)
33            query += " ";
34      }
35      console.log("retrieve & rank query: '" + query + "', collection: " + collectionName);
36      if (symptomList.length < 1)
37         console.log("Warning: request to Retrieve & Rank made with 0 symptoms!");
38
39      var fullString = 'https://091d880c-9617-490f-a859-87a7c7b1b8ad:IhxEV2KTiY1B@'
40         + 'gateway.watsonplatform.net/retrieve-and-rank/api/v1/solr_clusters/scc'
41         + 'aa3604c_1f02_4567_8162_c15dfe749fdf/solr/' + collectionName + '/fcsel'
42         + 'ect?ranker_id=' + rankerID +'&q='+ query +'?&wt=json&fl=id,title';
43
44      // perform the Retrieve & Rank API call:
45      var https = require('https');
46      https.get(fullString, function(resp){
47         var chunkText = '';
48         resp.on('data', function(chunk){
49            chunkText+=chunk.toString('utf8');
50         });
51         resp.on('end', function(){
52            var parseSuccess = Boolean(true);
53            var mJSON;
54            try {
55               mJSON = JSON.parse(chunkText);
56            } catch (err){
57               parseSuccess = Boolean(false);
58               console.log("Error parsing JSON response from Retrieve & Rank");
59               console.log(err.message);
60               console.log("Here is the R&R response that caused the problem:");
61               console.log(chunkText);
62            }
63            if (parseSuccess){
64               if (!mJSON.response)
65                  console.log("Error: the Retrieve & Rank HTTP request "
66                  +"did not produce a valid JSON");
67               else {
68                  if (mJSON.response.numFound == 0){
69                     console.log("Warning: the Retrieve & Rank request "
70                     + "returned 0 documents for the query");
71                     data.output.text = "We did not find any diseases "
72                     +"that were mentioned in conjunction with those symptoms.";
73                     return res.json(data);
74                  }
```

```
                    else {
                        var disorders = processJSON(mJSON);
                        var insertionMarker = "<insert parsed JSON response here>";
                        var currResponse = data.output.text;
                        var newResponse = currResponse.join('')
                            .replace(insertionMarker, disorders);
                        // make sure the substitution was made
                        if (currResponse === newResponse){
                            console.log("error - was unable to find marker '"
                            + insertionMarker + "' in the response");
                        }
                        else {
                            console.log("added the symptom list to the conversation "
                            +"response successfully");
                        }
                        // update the conversation response
                        data.output.text = newResponse;
                        return res.json(data);
                    }
                }
            }
        });
    });
    }
    // if there's no "no [more symptoms]" intent:
    else {

        // amusing responses to curse words:
        var text = "" + input.input.text;
        if (text.includes("Fuck") || text.includes(fuck") || text.includes("shit")){
            console.log("looks like the user just cursed");
            data.output.text = "It's going to be fine. Sooner or later, we all die :)";
        }
        else if (text.includes("ass")){
            data.output.text = "You and your physician may also want "
            +"to look into: Anger Management Disorders";
        }
        return res.json(data);
    }
}
```

2 Helper Functions

```
/** [EH] Gets a list of all the symptoms from the data object
 *    (returns an empty list if there aren't any)
 */
function extractSymptomList(data){
    var symptoms = new Array();
    for (var i = 0; i < data.entities.length; i++) {
```

```
7         symptoms.push(data.entities[i].value);
8      }
9      return symptoms;
10  }
11
12  /** [EH] Helper method for removing duplicates from a list
13   */
14  function removeDuplicates(list){
15     var listNoDuplicates = new Array();
16     for (var i = 0; i < list.length; i++){
17        var inListAlready = Boolean(false);
18        for (var j = 0; j < listNoDuplicates.length; j++){
19           if (list[i] === listNoDuplicates[j]){
20              inListAlready = Boolean(true);
21           }
22        }
23        if (!inListAlready){
24           listNoDuplicates.push(list[i]);
25        }
26     }
27     return listNoDuplicates;
28  }
29
30  /** [EH] Given a "data" JSON object, checks whether its intent list
31   *     contains an intent of the given name
32   */
33  function hasIntent(data, intentString){
34     if (data.intents){
35        for (var i = 0; i < data.intents.length; i++){
36           if (data.intents[i].intent === intentString){
37              return true;
38           }
39        }
40        return false;
41     }
42     else {
43        console.log("hasIntent error - data object has no intent field");
44     }
45  }
46
47  /** [EH] Given a valid JSON response from Retrieve & Rank, extract
48   *     the important information.
49   */
50   function processJSON(json){
51     console.log("---processing json response: "
52        + json.response.numFound + " documents returned");
53     var titleList = new Array();
54     for (var i = 0; i < json.response.docs.length; i++){
55        titleList.push("" + json.response.docs[i].title);
56     }
57     var disorderList = removeDuplicates(titleList);
58     var listString = "";
```

```
59    for (var i = 0; i < disorderList.length; i++){
60       listString += " " + disorderList[i];
61       if (i < disorderList.length - 2)
62          listString += ",";
63       else if (i == disorderList.length - 2){ // after second-to-last item
64          if (i == 0){ // two-item list
65             listString += " and";
66          }
67          else if (i > 0){
68             listString += ", and";
69          }
70       }
71    }
72    return listString;
73 }
```