**Shilpa Murali**

**Similar Doctors**

**Output:**

The output is provided as a JSON object which can be used to provide as an input to the fron end framework and display in the website.

{

"00010": [

{

"area": "Santa Clara",

"doctorID": "00010",

"experience": "15",

"name": "sjaa",

"review": 3.5,

"school": "Havard",

"speciality": "Ortho"

}

],

"10002": [

{

"area": "MV",

"doctorID": "10002",

"experience": "1",

"name": "bMohan",

"review": 3.0,

"school": "Havard",

"speciality": "Ortho"

}

]

}

**Future Work:**

* In an actual implementation, the design of back end for this system can be done using two tables in a MongoDB database.

1. Collection "doctors" that contains the following document as key vale pairs:

{"doctorID":String; "name":String; "speciality":String; "insurance":Array,"experience":String,"languages":Array,"school":String}

2. Collection "reviews" that contains the reviews that are associated with a doctor as key vale pairs:

{"doctorID":String;"review":Array}

* The parameters like the list of Insurances covered by each doctor, the languages spoken by the doctor can be added as attributes to measure the similarity. These parameters will be in the form of lists and the similarity can be calculated by comparing the elements in both the lists and finding a ratio of similar elements/ total elements and proving a proportional similarity score
* The review score similarity can be modified by including those reviews which are above a threshold value. For eg, if the doctor input has a review score of 3.5 we can find doctors whose difference of review score with the doctor input is positive (eg, 4.0,4.5 and not 3.2,3.1 etc) and include doctors whose reviews score is above a threshold from the review score of the doctor input. This is because it will be better if the system recommends doctors with higher reviews.
* The Area of the doctor can be segregated as zones using the distance in miles. This can be calculated using the zip code of the area and provide scores proportional to the distance in miles from the area of the input doctor.
* The computation of the similarity metrics is done offline. This is because the details of the doctors do not change so frequently. So when the user requests to find the similar doctors, we can use the pre computed similarities.
* This will avoid the latency and load in computing the similarity in real time. Also if new doctors are included, we have to calculate the similarity of the new doctor with all other doctors and add the new doctor into the heapq of other doctors of their similarity level is greater than the min value in the heapq. This becomes and O(1) operation.