**QB Skills Based Model**

The Skill Based Model aims at generating a questionnaire based on the details provided by the user.

The Code is elaborated in an extensive way as follows:

**Input**

The code takes input in the form of a json file. This file must contain every feature the user wants to put in, in a detailed format:-

{

"\_id":{"$oid":"5ecea1b690cfa44a44f83139"},

"user\_id":"Caramel\_1",

"firstName":"Student",

"lastName":"One",

"emailAddress":"student1@mail.com",

"category":"Student",

"password":"$2a$10$xdzs2VniPSbpZ0sxvpoq2OEQTcOx0LbsroyNWEHVqKnnOrM.qbj8i",

"mobileNumber":6546546541,

"state":"State",

"collegeName":"College",

"total\_skills":4,

"skill\_name1":"Html",

"skill\_name2":"Css",

"skill\_name3":"JS",

"skill\_name4":"Bootstrap",

"skill\_name5":"",

"skill\_prof":"4",

"saltSecret":"$2a$10$xdzs2VniPSbpZ0sxvpoq2O",

"\_\_v":0

}

The User has to mention the Skills that he has along with his proficiency in them.

\*In my analysis, *for this specific model*, I’ve noticed than many of the details present in the Json input file hinders the user’s privacy. So, it is able to remove many details like Email Address, Password, and Mobile Number. It still won’t affect the flow of the program.

Other than this the user has the option to choose how many more question he wants to answer. This option is asked separately.

**Code**

* Every user has to go through 10 specific and generic questions. \*They can add more to the list of questions and would be asked to do so at the start.
* Data is taken from the json input like ID, Skills, and category. Now taking in consideration the User Proficiency and Category provided, the difficulty of the user is set as medium or hard. The default setting is to keep it easy.
* The getmcq(skill, difficulty, id2, no\_of\_questions) function is called which take in account the skills, difficulty id and the no of questions the user wants. The function is divided into 2 parts. Since the questions to be given have to be equally distributed among the skills provided by the user. Therefore no of questions should be easily divisible by the no of skills provided (where maximum is 5). So the code divides the questions between prime (1, 2, 3, 5) and non-prime (4) divisible parts.

Example: for 30(number) questions if 1,2,3,5 number of skills are entered then no of questions will be divided equally among the skills

Example: for 4 skills and 30 questions. This will be for if user enters 4 skills then 2 skills will have 8 questions each and 2 will have 7 questions

* Data is imported into a dataframe (df) through Dataframe Methods present in the pandas library. This dataframe is further processed and filtered on the basis of difficulty and the topic. It shuffles randomly by the sample method provided by pandas. 10 generic topic questions are then added to the dataframe in a shuffled manner.
* Finally the QIDs of the questions are extracted into a final dataframe. \*Previously this dataframe was indexed. I suppose due to the use of a NoSQL database. But if Structured Sql is used then indexing is not required

**Output**

The final dataframe is then converted to json format and saved into a separate file with the name of the user id.

Example output (where there are total 15 questions = generic (10) +NoOfQuestions (5))

{"QID":{ "0":"19PYM","1":"25PYM",

"2":"13PYM","3":"2JSM",

"4":"1PYM","5":"4GEM",

"6":"2GEM","7":"5GEM",

"8":"6GEM","9":"11GEM",

"10":"6GEM","11":"8GEM",

"12":"1GEM","13":"10GEM",

"14":"8GEM"

} }

\*Previous Json ouput had two coloums which not only adds significant overhead over data transfer but also is useless in cases of structured databases.