Software Design Specification for the Insticator questionnaire web service

YIze Liu, Boston University, lyz95222@bu.edu

1. Overview

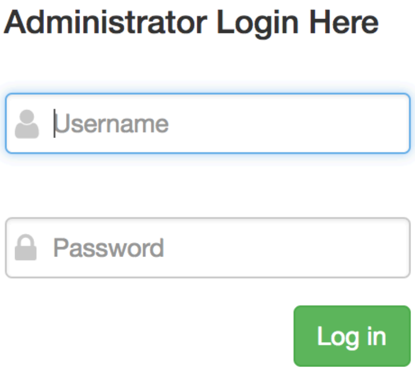
This web service contains two parts:

1. A backstage management system for administrator to manage all the questions, including add, delete, select and update operations for each question.
2. A frontend questionnaire web service to show all kinds of questions to the users, including trivia (single choice question with one current answer), poll (objective single choice question), checkbox (objective question with multiple choices) and matric question (objective question that be displayed as a matric)

2. Function Flow

(1): Management system:

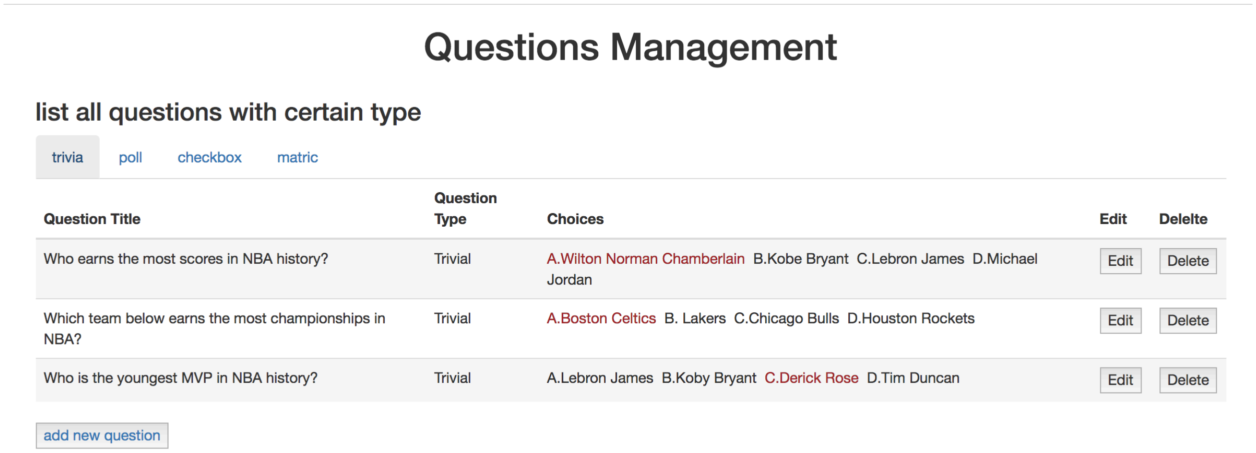
Administrator log in
The administrator could login with pre-registered account and password (I didn’t implement the registration page for the administrator due to the consideration of the security).



Administrator Login Here

Query all kinds of questions
Once the administrator login into the system, he/she could view all kinds of questions by select the question type on the navigation bar:

Eg1: by clicking the trivia button, all trivia questions could be shown below:



Questions Management

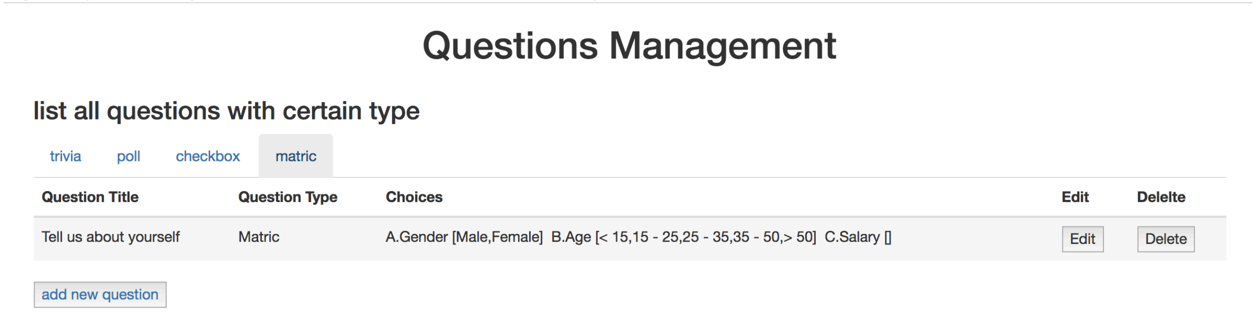
list all questions with certain type

Question Title	Question Type	Choices	Edit	Delelte
Who earns the most scores in NBA history?	Trivial	A.Wilton Norman Chamberlain B.Kobe Bryant C.Lebron James D.Michael Jordan	Edit	Delete
Which team below earns the most championships in NBA?	Trivial	A.Boston Celtics B. Lakers C.Chicago Bulls D.Houston Rockets	Edit	Delete
Who is the youngest MVP in NBA history?	Trivial	A.Lebron James B.Koby Bryant C.Derick Rose D.Tim Duncan	Edit	Delete

Figure 1. select all trivia questions

Notice that all trivia questions require a right answer and the right choice is set as red color in our website.

Eg2: by clicking the matric button, all matric questions could be shown below:



Questions Management

list all questions with certain type

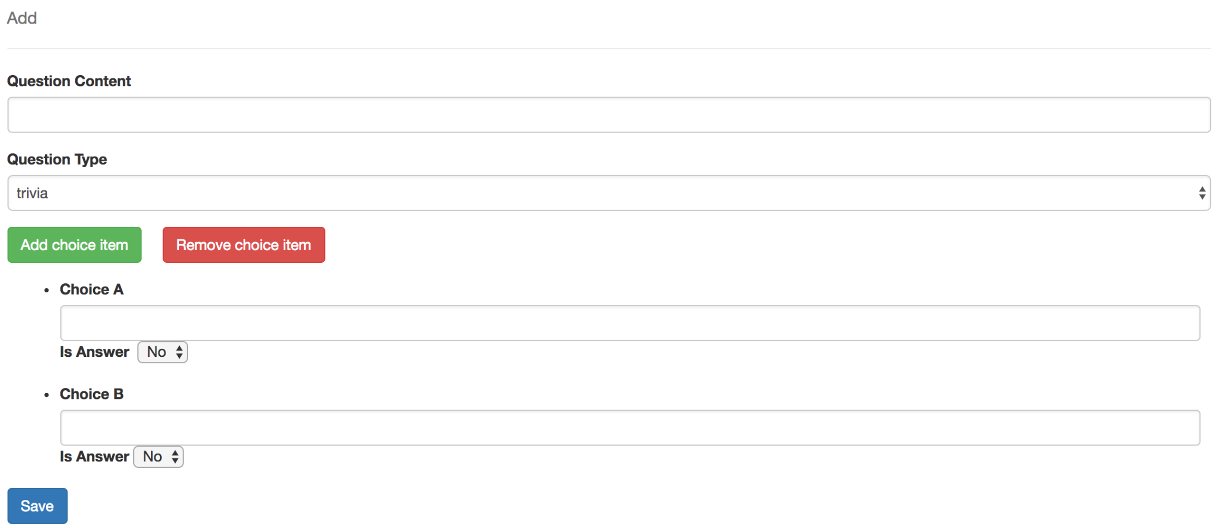
Question Title	Question Type	Choices	Edit	Delelte
Tell us about yourself	Matric	A.Gender [Male,Female] B.Age [< 15,15 - 25,25 - 35,35 - 50,> 50] C.Salary []	Edit	Delete

Figure 2. select all matric questions

Since each matric includes one row and one column, each of which contains several default choices and they are shown in an "[]" array-like structure behind each choice content.

Add new question

When the administrator click the ‘add new question’ button below, it will enter into the page for adding question.


Add

Question Content

Question Type: trivia

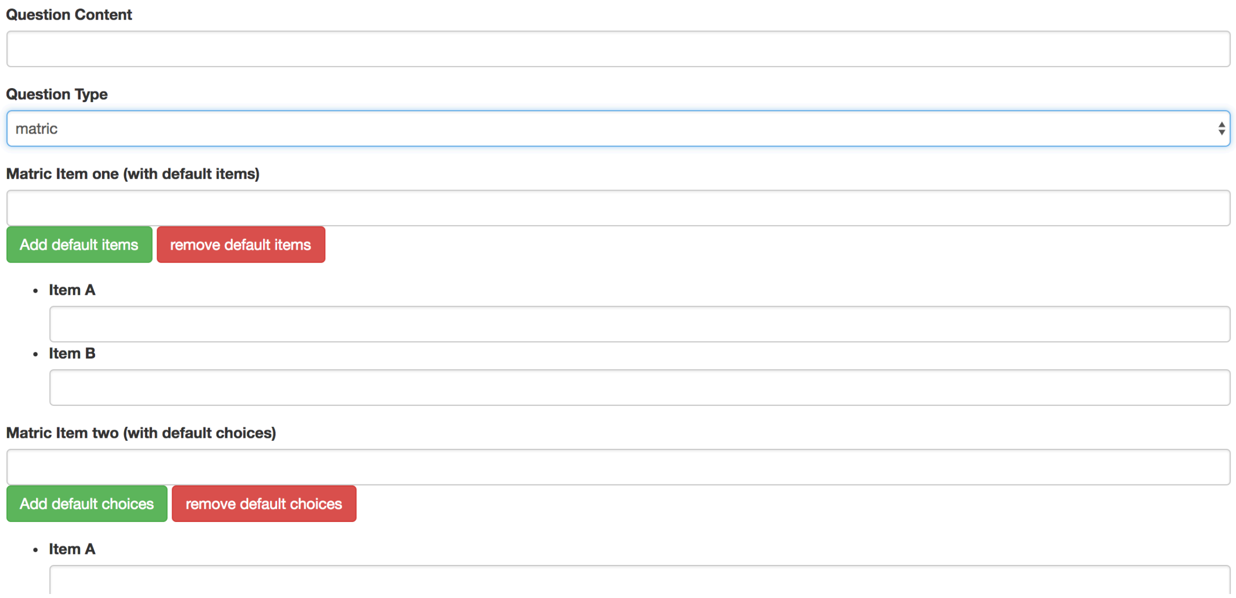
- Choice A

  Is Answer: No

- Choice B

  Is Answer: No

Figure 3. add a new trivia question


Question Content

Question Type: matric

Matric Item one (with default items)

- Item A
- Item B

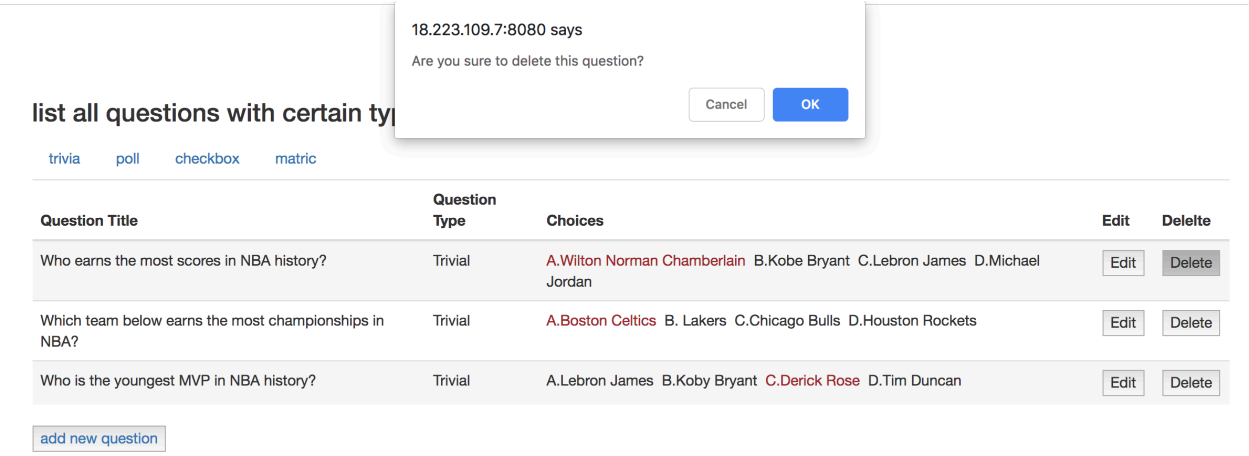
Matric Item two (with default choices)

- Item A

Figure 4. add a new matric question

By adding a new question, you could click add choice items and determine which one is the right answer. The layout for adding the matric item is somehow different with other question type since we need to specify the cols and rows title for each matric. Also, the checking condition for each type of question is also somehow different. By clicking the save button at the bottom, you could add a new question to the database if you fill out all the required information.

Delete a question



18.223.109.7:8080 says

Are you sure to delete this question?

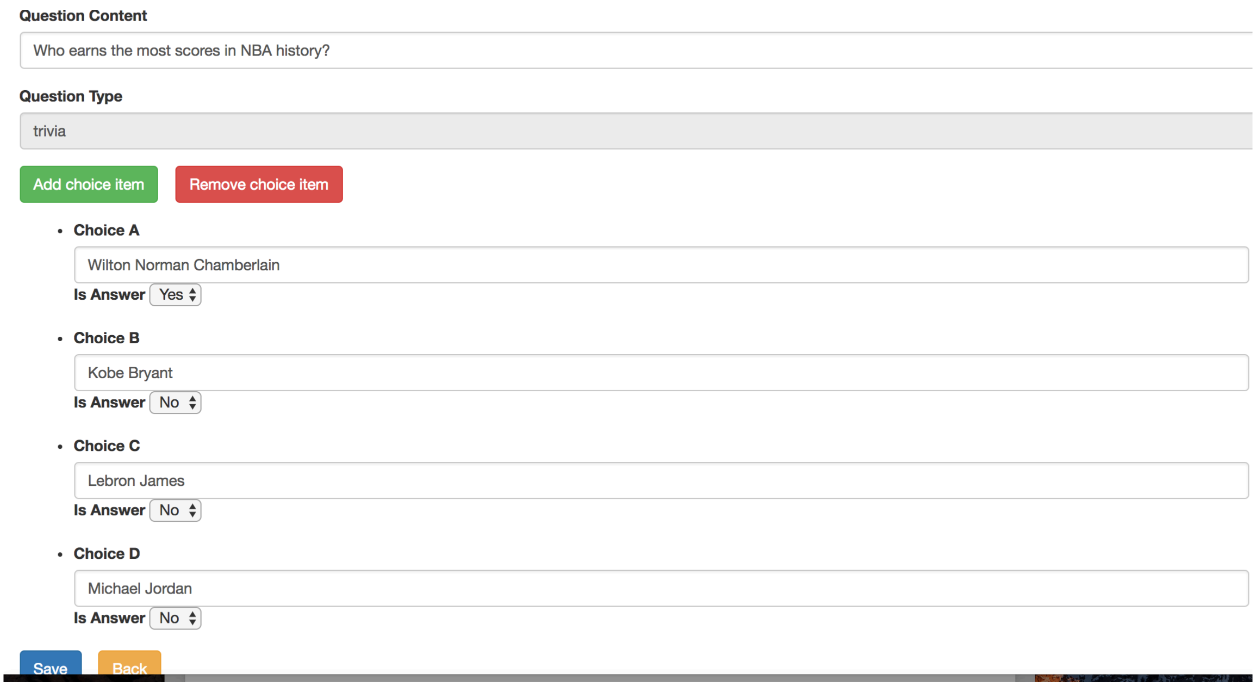
list all questions with certain typ

Question Title	Question Type	Choices	Edit	Delelte
Who earns the most scores in NBA history?	Trivial	A.Wilton Norman Chamberlain B.Kobe Bryant C.Lebron James D.Michael Jordan	Edit	Delete
Which team below earns the most championships in NBA?	Trivial	A.Boston Celtics B. Lakers C.Chicago Bulls D.Houston Rockets	Edit	Delete
Who is the youngest MVP in NBA history?	Trivial	A.Lebron James B.Koby Bryant C.Derick Rose D.Tim Duncan	Edit	Delete

Figure 5. delete question

By clicking the 'Delete' button on the rightmost of each row. We could remove that question from our database.

Edit the question



Question Content

Who earns the most scores in NBA history?

Question Type

trivia

- Choice A

  Wilton Norman Chamberlain

  Is Answer Yes

- Choice B

  Kobe Bryant

  Is Answer No

- Choice C

  Lebron James

  Is Answer No

- Choice D

  Michael Jordan

  Is Answer No

Figure 6. edit question

By clicking the edit button for each question, we could enter into the edit page with all the question info listed. By clicking the save button, you could update the information of the question.

(2): User questionnaire system:

User login and register

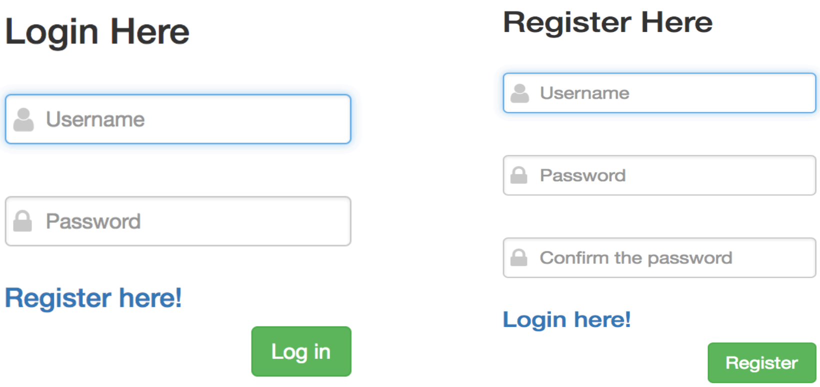
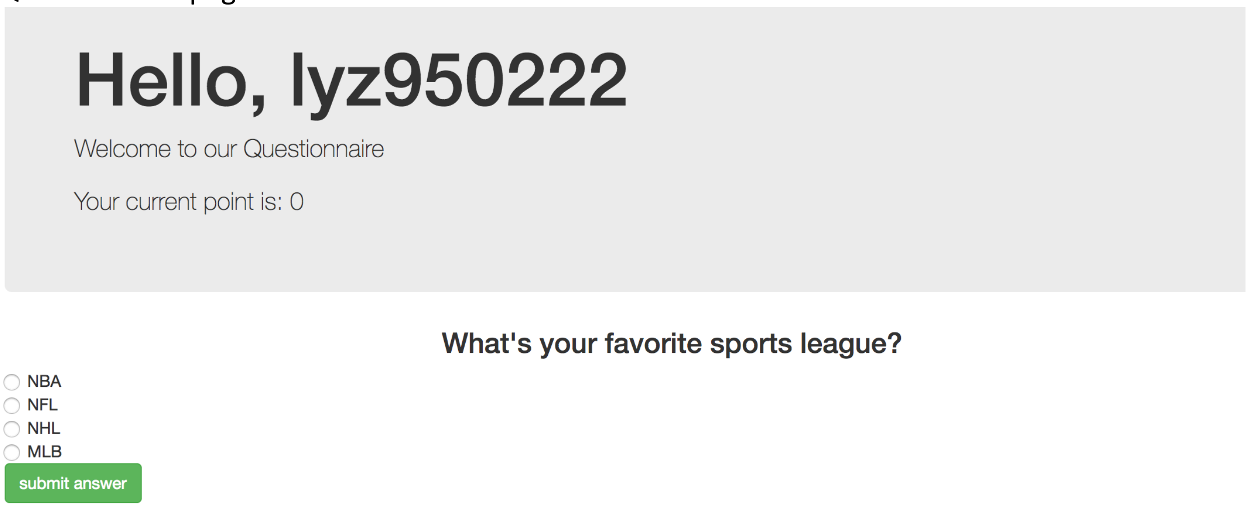


Figure 7. user login and signup

User could register an account and login into the system.

Questionnaire page



Hello, lyz950222

Welcome to our Questionnaire

Your current point is: 0

What's your favorite sports league?

○ NBA
○ NFL
○ NHL
○ MLB

The Questionnaire page will show username and current points he/she earned. The layout not looks good now due to the time limit. However, the function works well.

3. Software Design

(1): Database:

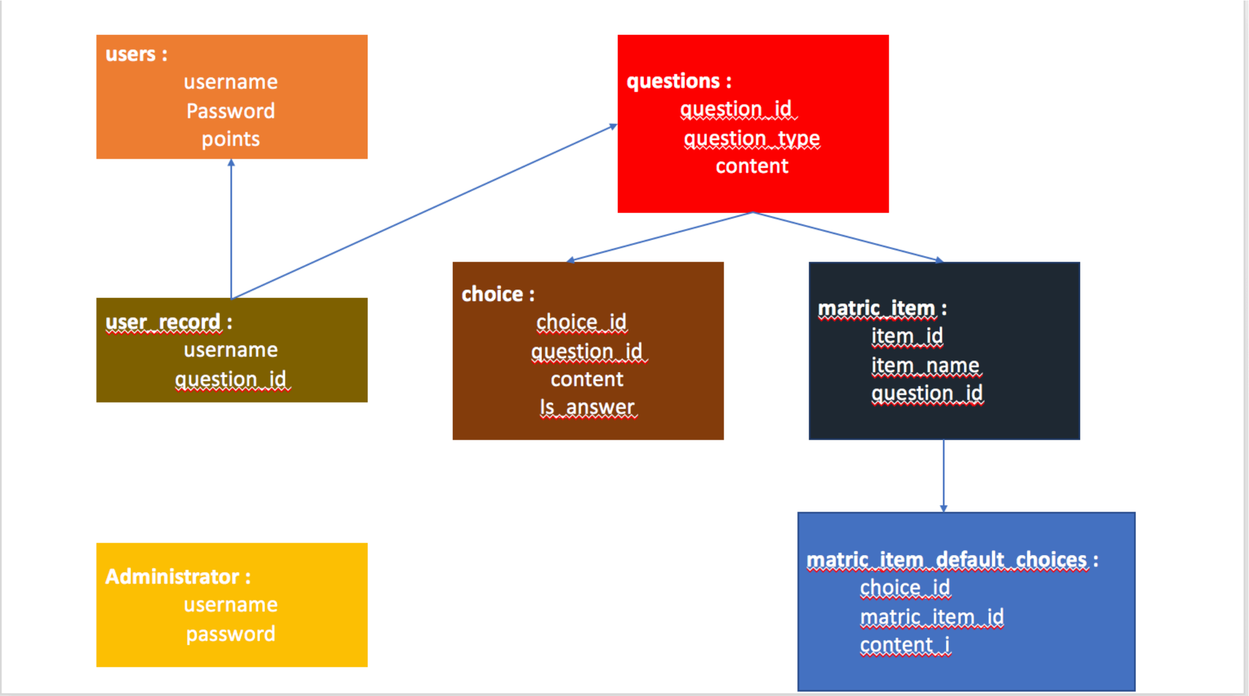


Figure 8. database structure

I choose Mysql as the database service. The structure of the system's database structure is shown in the figure above. This system includes seven tables:

(1) administrator: record the username and password of the administrator who has the right to operate questions.

(2) users: record the information of each user, include their username, password and points they earned.

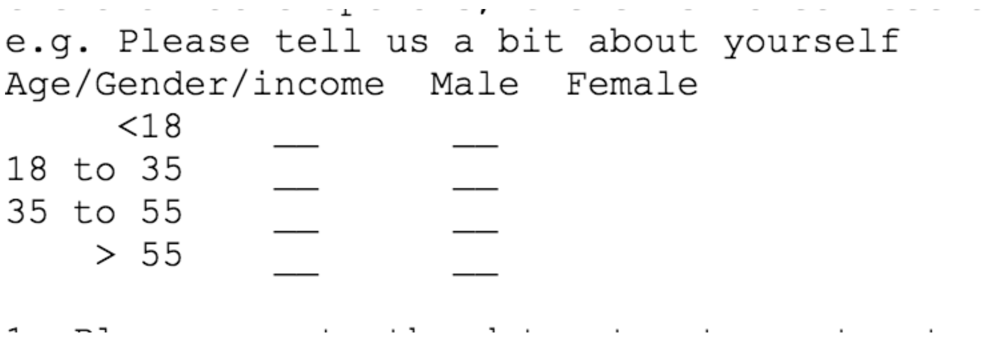
(3) questions: record the info of each question, which contains its id, the type (we save each type as an int value in our database, the relations are: 0: trivia; 1: poll; 2: checkbox, 3: matric). And also content to record the title of the question.

(4) user_record: when user finish answer a question, in order to keep the same question showing to this user again, we need to record all the questions the user have asked by using this table.

(5) choice & matric_item: when the type of question is poll, trivia and checkbox. We save the choices in the choice table, which includes the id of the question it points to and the choice content. Also, we add a Boolean field called isAnswer to check whether such choice is a right answer. For objective questions, such attribute will be set as false by default.
For matric question, we use matric_item table to record its conten in row & col and use matric_item_default_choice to record the default choice items of each matric item.

For example:

For a matric question below:



```
e.g. Please tell us a bit about yourself
Age/Gender/income  Male  Female
     <18            __     __
18 to 35            __     __
35 to 55            __     __
    > 55            __     __
```

In this case, we save this question in three steps:
1st: Insert a new item into the question table with:

question_id: 1
question_type: 3
question_content: ‘Please tell us a bit about your self’

2nd: Insert three new matric items into matric_items table

{ item_id: 0 question_id: 1 item_name: ‘Age’}
{ item_id: 1 question_id: 1 item_name: ‘Gender’}
{ item_id: 2 question_id: 1 item_name: ‘income’}

3rd: Insert new items into matric_item_default_choices table

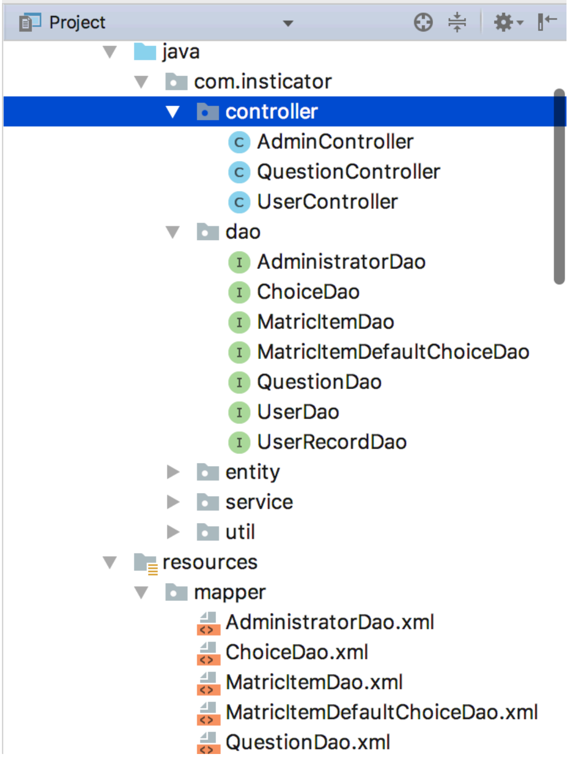
For Age matric item:
{choice_id: 0, matric_item_id: 0, content: ‘<18’}
{choice_id: 1, matric_item_id: 0, content: ’18 to 35’}
{choice_id: 2, matric_item_id: 0, content: ’35 to 55’}
{choice_id: 3, matric_item_id: 0, content: ‘>55’}

For Gender item:

{choice_id: 4, matric_item_id: 0, content: ‘Male}
{choice_id: 5, matric_item_id: 0, content: ’Female’}

(2): MVC design:

The structure of the whole project is kind of similar as MVC prototype, which are shown below:



```
Project
  java
    com.insticator
      controller
        AdminController
        QuestionController
        UserController
      dao
        AdministratorDao
        ChoiceDao
        MatricItemDao
        MatricItemDefaultChoiceDao
        QuestionDao
        UserDao
        UserRecordDao
      entity
      service
      util
  resources
    mapper
      AdministratorDao.xml
      ChoiceDao.xml
      MatricItemDao.xml
      MatricItemDefaultChoiceDao.xml
      QuestionDao.xml
```

I split the whole project into four layers: dao (model) – service – controller (control) – frontend page (view)

Dao (model layer):

In this layer, I created java pojo, which are all included in package entity, for each related database table. And use mybatis to de-couple the java codes and the sql codes. All the methods that are interacted with the Mysql database are all defined in the dao packages by using java interface. And those methods are implemented in recources/mapper directory by using xml files.

Service (intermediate layer):

I include the service layer to act as an intermediate layer, which supports some complicated sql update and insert functions and also some transactional operations.

Controller (controller layer):

In this layer, I provide the interface for frontend web pages to pass and transform data. The controller includes three parts: userController (handle all requests from users), adminController (handle requests from administrator) and quetionController (provide methods for handling question CRUD calls).

4. Drawbacks and Improvements

Due to the time limit, this system is not totally complete and here are some drawbacks about this service.
First, we do not record user's choice for each question. We just add user's points when they reach the right answer. However, if the user wants to trace his history, we could not provide this service.
Also, the front-end page, especially for the user questionnaire page still not looks good enough, which may need to polish later.

5. How to make the user requests scalable

1: Changing the mysql server structure by using a master-slave structure. We could use one master database where data is written to and it is replicated to multiple slave servers. Since in our system, the write operations is quite less than reads because we only let one administrator to update questions but could allow multiple users to query the questions. Thus, such structure could perform faster when handling large amounts of queries.
2: Using caching database: to accelerate the reading speed of the database. We could add a NoSQL database acted as a cache to increase the query speed (like redis).
3: By using multiple servers and Load Balancer: If we have millions of requests, it's impossible to handle all the requests by just using one server. Therefore, we could use Load Balancer to contain multiple web servers and forward incoming requests to one of them to distribute.