SINGAPORE UNIVERSITY OF SOCIAL SCIENCES

ICT239
Web Application Development

Tutor-Marked Assignment
January 2023 Presentation

TUTOR-MARKED ASSIGNMENT (TMA)

This assignment is worth 24 % of the final mark for ICT239 - Web Application Development

The cut-off date for this assignment is **Monday, 1st May 2023, 2355 hours.**

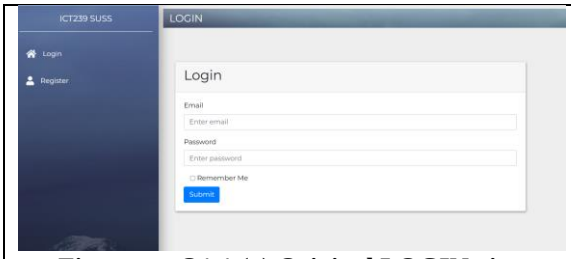
Note to Students:
You MUST **use the provided solution template** accompanying this TMA.

Answer all questions. (Total 100 marks)

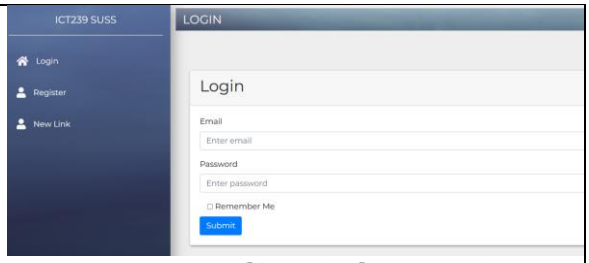
Question 1 (40 marks)

Question Q1 concerns the BMI case study.

(a) In the original view for Login as shown in Figure Q1.1 (a), both the URL links at the sidebar and Login Card view will change when hovered over.



Figureure Q1.1 (a) Original LOGIN view



Figureure Q1.1 (b) Revised LOGIN view

Revise the source code in ONE (1) HTML and ONE (1) CSS file so that

(i) A new URL link as shown in Figure. Q1.1 (b), similarly shaped, is added to the sidebar. Name the HTML file changed.

(ii) The links and the Login card should not change when hovered over. Name the CSS file changed.

(2 marks)

(b) Describe what happened to cause the transition from the LOGIN view in Figureure. Q1.1 (a) to the DASHBOARD view in Figureure Q1.2 (a) and then to the LOGGING BMI 2 view in Figureure Q1.2 (b).

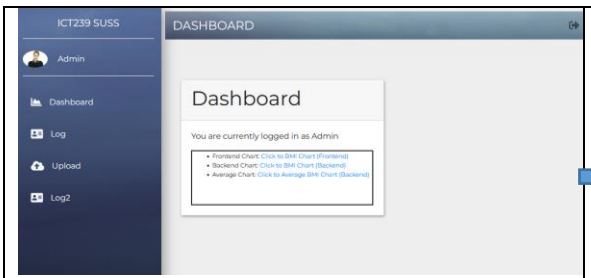


Figure. Q1.2 (a) DASHBOARD view

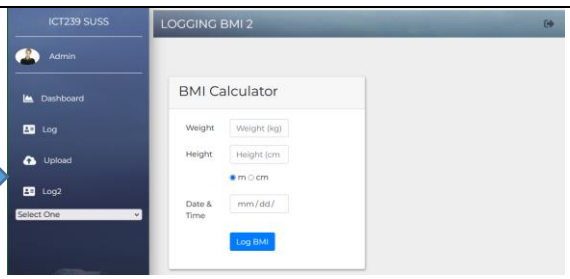


Figure. Q1.2 (b) LOGGING BMI 2 view

(i) Tracing: What user actions will cause the LOGIN view in Figure Q1.1 (a) to transit to the DASHBOARD view in Figure. Q1.2 (a) and the DASHBOARD view in Figure. Q1.2 (a) to transit to the LOGGING BMI 2 view in Figure. Q1.2 (b)?

(ii) Flask routing: Which Flask routes and the associate HTTP requests and responses are involved in executing the transitions?

(iii) Jinja: Describe the Input-Processing-Output (IPO) in the Jinja execution that help render the LOGGING BMI 2 view in Figure. Q1.2 (b) when an appropriate user action is applied on the DASHBOARD view.

(iv) Flask Blueprint: Name the source code files and point out the statement(s) where Flask Blueprint functionality is employed in the transition from Figure. Q1.2 (a) to Figure. Q1.2 (b). What does that functionality achieve in the transition?

(12 mark)

(c) For the Registration and Login functions of the BMI case study,

(i) Simulate **ONE (1)** failed session for EACH of the two functions.

(ii) For each of the above **TWO (2)** sessions, revise the source codes so a different error message with similar meaning is displayed.

(iii) Simulate the same **ONE (1)** failed session for EACH of the two functions after you have revised the source code.

(5 mark)

(d) Describe the context before and after the Submit button is successfully clicked in the LOGIN BMI 2 view in Figure. Q1.3 (b).

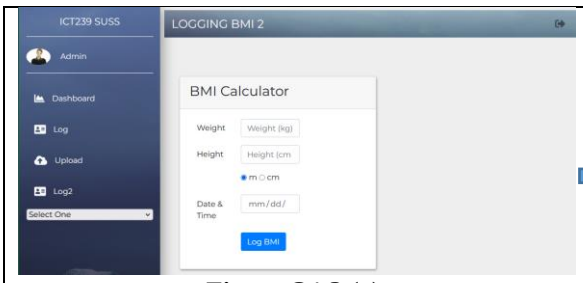


Figure Q1.3 (a)

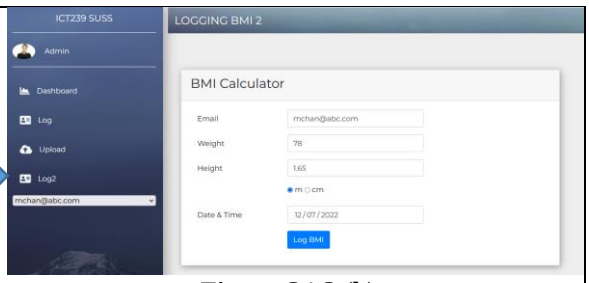


Figure Q1.3 (b)

(i) Jinga and JavaScript:

- Identify the file which contains statements(s) to provide values for populating the available choices in the dropdown list in the sidebar in Figure. Q1.3 (a). Which file uses the values and explain how the dropdown list is being populated.

- The transition from Figure. Q1.3(a) to Figure. Q1.3(b)

Describe the Event Management mechanism implemented via JavaScript, namely (1) which Event on (2) which Element is monitored and (3) what is the response to the Event happening.

Identify the file where the relevant code is found and used in your description.

(ii) Flask routing:

- Which Flask route and the associated HTTP request are involved when the Log BMI button is clicked in Figure. Q1.3(b)?

- Explain what the execution of the view function achieved. What is the Http response?

(iii) Model:

- Which models store the data/results when the Submit button is clicked in Figure. Q1.3(b)?

- List TWO blocks of MongoEngine code that define the models.

(10 marks)

(e) Describe what happened when the "Click to BMI chart (backend)" link is clicked to cause the following transition from Figure. 1.4(a) to Figure. 1.4(b).

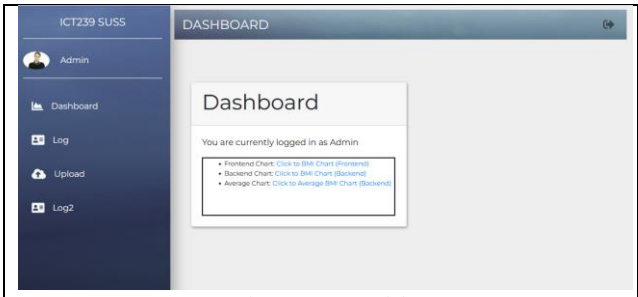


Figure. Q1.4 (a)

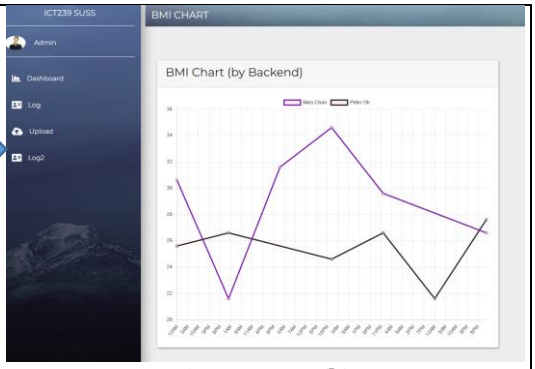


Figure. Q1.4 (b)

Flask routing and AJAX:

(i) Which Flask route and the associate HTTP request and response are involved in the transition from Figure. 1.4 (a) to Figure. 1.4 (b) before the AJAX call was made? Explain what the execution of the view function achieved.

(ii) Describe the AJAX call and its purpose.

(iii) Which Flask route and the associate HTTP request and response are involved in the transition from Figure. 1.4 (a) to Figure. 1.4 (b) upon making the AJAX call and when the call returns? Explain what the execution of the view function achieved and how the chart was generated.

(11 marks)

Questions Q2 – Q4 concern the development of a Web application for managing the golf sets of members of a golf club. The application scenario is based on that of the ICT162 July 2022 semester TMA.

Question 2 (32 marks)

Learning objectives:

- Apply programming methods to present information in HTML
- Employ web programming framework for developing website
- Construct a prototype website to present information from multiple sources upon user requests.

(a) Register, Login and Logout

Refer to Figure. 2.1 (a) – Figure. 2.1 (c) for the user interfaces for Register, Login and Logout. Similar to the BMI case study, before a user can use the application, he must register for an account. To exit from the application the user selects log out. The log out functionality is available only after a successful login.

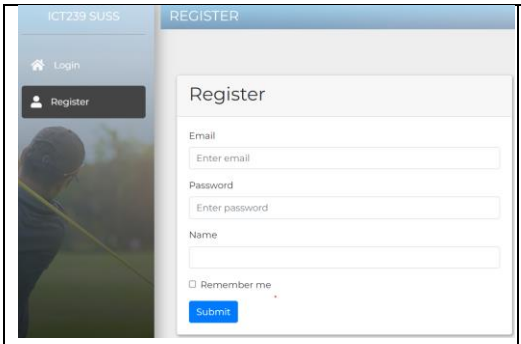


Figure. Q2.1 (a)

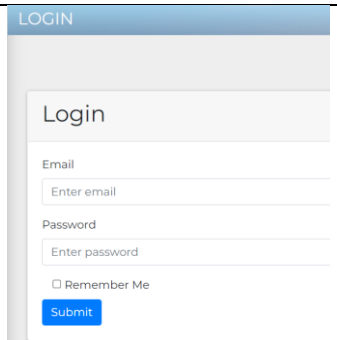


Figure. Q2.1 (b)

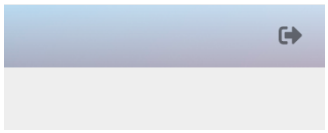


Figure. Q2.1 (c)

However, in the new application, after the user logins in successfully, he is directed to a page as shown in Figure Q2.2 (a) if logged in as admin user (admin@abc.com), and as shown in Figure. Q2.2 (b) if logged in as a non-admin user.

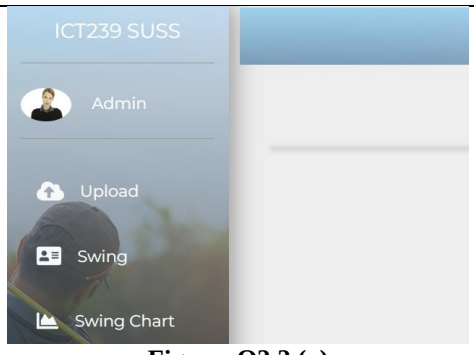


Figure. Q2.2 (a)

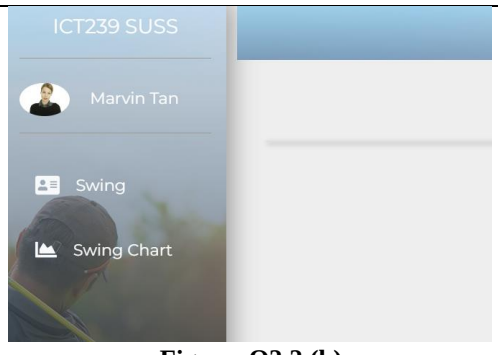


Figure. Q2.2 (b)

Modify the code in the BMI case study to achieve the new effects. Explain your modification. (4 marks)

(b) Upload files

The Upload function is available to only the admin user. To access the function, the user clicks on the Upload link in the sidebar. There are two dropdown lists: one to select a registered user as shown in Figure. 2.3 (a), and another to select the type of file to upload as shown in Figure. 2.3 (b). Once a file is chosen, the admin user clicks on Upload to activate the upload of data to the backend to be stored in Mongodb.

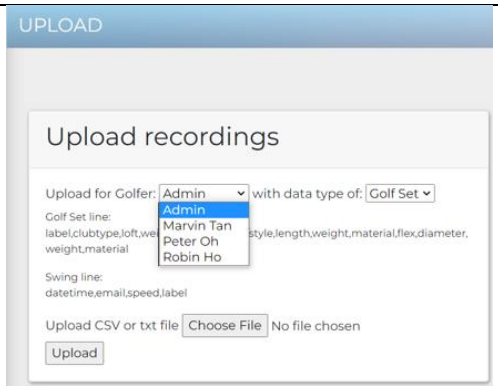


Figure. Q2.3 (a)

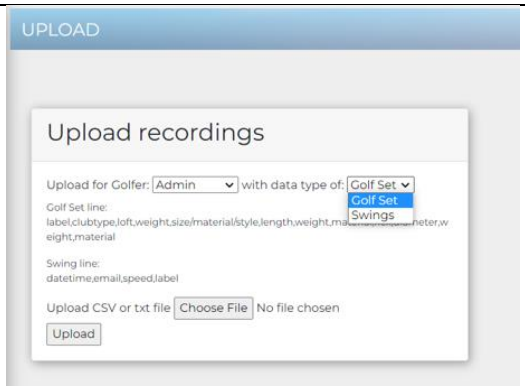


Figure. Q2.3 (b)

Golf Set file

The Golf Set file is a text files which contain details of clubs in a golf set for a registered user. An example file "A20-Marvin.txt" for a registered user e.g., Marvin Tan is reproduced here:

"A20-Marvin.txt"

```
DRIVER,Wood,10.5,203,450,44.25,68,Graphite,R,0.6,62,Rubber
5-WOOD,Wood,17.5,240,280,41.5,85,Graphite,R,0.6,62,Rubber
5-IRON,Iron,26.5,262,Cast,38.0,102,Steel,S,0.6,62,Rubber
6-IRON,Iron,30.0,264,Cast,37.5,104,Steel,S,0.6,62,Rubber
7-IRON,Iron,33.5,266,Cast,37.0,106,Steel,S,0.6,62,Rubber
8-IRON,Iron,37.0,268,Cast,36.5,108,Steel,S,0.6,62,Rubber
9-IRON,Iron,40.5,270,Cast,36.0,110,Steel,S,0.6,62,Rubber
PW,Iron,45.0,270,Cast,35.75,112,Steel,S,0.6,62,Rubber
GW,Iron,50.0,270,Cast,35.5,113,Steel,S,0.6,62,Rubber
SW,Iron,54.0,270,Cast,35.25,114,Steel,S,0.6,62,Rubber
LW,Iron,58.0,270,Cast,35.0,115,Steel,S,0.6,62,Rubber
ODYSSEY#7,Putter,3.0,365,Mallet,34.0,120,Steel,S,0.6,62,Rubber
```

The format of each line in the text file, as given in the appendix of ICT162 TMA are as follows:

<label>,<clubtype>,<loft>,<weight>,<size/material/style>,<length>,<weight>,<material>,<flex>,<diameter>,<weight>,<material>

where highlighted in yellow is ClubHead information, in green is Shaft information, and in blue is Grip information. Notice that a club consists of a club head, a shaft and a grip as shown in Figure. 2.4. You can assume that there are no two identical clubs in a golf set belonging to one user, Marvin Tan with email marvin@abc.com in this example. Furthermore, a club is identified by the club label.

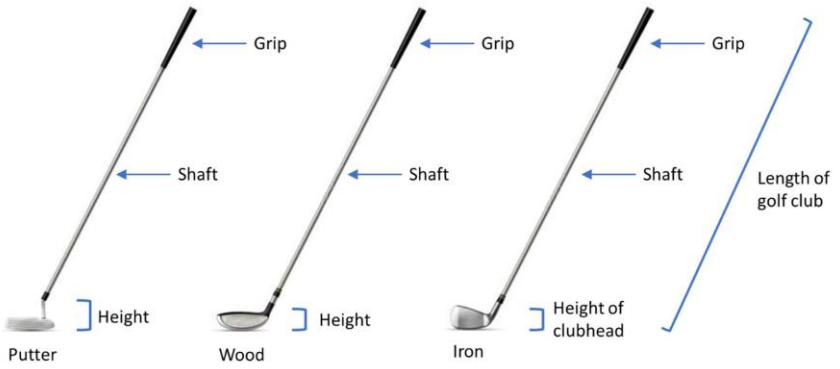


Figure. Q2.4

Note: <size/material/style> depends on <clubtype>.The application will consider only these three types of club heads: Putter head, Wood head and Iron head. Refer to Figure. Q2.5 for details on the club heads.

Putter	Wood	Iron
		
Putter heads has different styles like blade, half-mallet, and mallet. The putter is used to roll the ball forward into the hole.	Woods are bigger and rounder clubheads that are designed to shoot the ball over long distances. The size of the head is measured in cubic centimetre (cc).	Made of hard (cast) or soft (forged) iron. Irons are the most represented clubs in a golfer's bag, numbering from 3-iron to 9-iron, pitching and sand wedge.

Figure. Q2.5

Refer to https://sites.google.com/site/learngolfbasics/golf-kit/club-loft-distance for an introduction to golf basics.

Swings file

The Swing file is a csv file which contains details of swings made by a registered golfer. An example file "swing-Marvin.csv" for a registered user, Marvin Tan, is reproduced here:

```
swing_time,swing_speed,club_label
"""2023-3-10T07:30""",80,DRIVER
"""2023-3-10T07:32""",85,5-IRON
"""2023-3-10T07:35""",95,DRIVER
"""2023-3-10T07:40""",98,5-IRON
"""2023-3-10T07:44""",90,5-IRON
"""2023-3-10T07:47""",95,5-IRON
```

Each line in the file records the time the swing was made, the speed of the swing in miles per hour (mph) and the club specified by club label, used to make the swing. The estimated distance in yards, travelled by the golf ball when a swing is made is computed using the formula:

```
estimated_distance = (280 - abs(48- club_length)*10 - abs(club_head_loft
- 10)*1.25) * swingSpeed/96

club_length = club_head_height + shaft_length

club_head_height =  club_head_size/400 for WoodHead
                    1 for IronHead
                    1 for Blade style and 0.5 otherwise for PutterHead
```

The class diagram is shown in Figure. Q2.6.

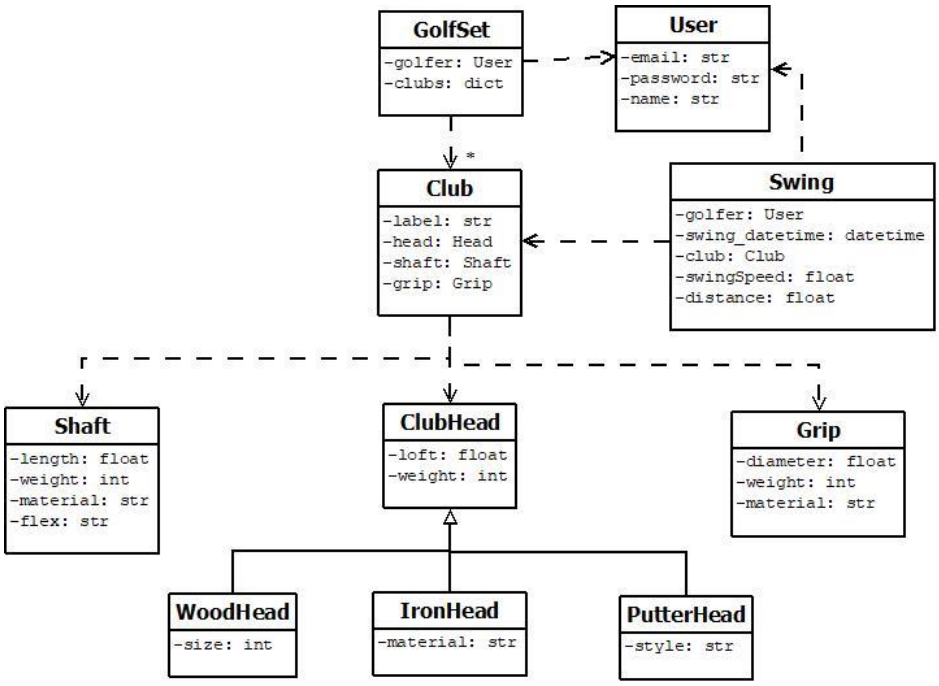


Figure. Q2.6

You may implement the model component with any number of collections and appropriate document structures using mongoengine and mongodb. Include supporting functions/methods where applicable into the python module(s) for your model component so that view functions will not access the database directly but will call the supporting functions/methods.

Additional 5 bonus marks will be given to you if you apply object-oriented principles correctly in your implementation for (i) to (iii), in particular, if you use object composition (using object references) and inheritance based on Figure. Q2.6.

(i) Implement the model(s) to store the content of the uploaded text files for the clubs in a golf set for a registered golfer, including the supporting functions/methods. Provide an explanation for your class you define.

(10 marks)

(ii) Implement the model(s) to store the content of the uploaded csv file for the swings made by a registered golfer, including the supporting functions/methods. The estimated distance should be computed and stored with each swing.

(4 marks)

(iii) Implement the frontend and backend components for the Upload function for both types of files. You should handle erroneous situations such as when a club cannot be located in a golf set of a golfer when recording his swing.

(12 marks)

Question 3 (20 marks)

Learning objectives:

- Employ web programming framework for developing website
- Construct a prototype website to present information from multiple sources upon user requests.

Question 3 is an extension of Question 2 where you are to develop the SWING function to allow user to enter swing data via a form instead of through file upload.

As explained in Question 2, depending on which user is accessing, the application displays two different sidebars: the admin user has an additional Upload link followed by the Swing link as shown in Figure. Q3.1 (a) whereas for non-admin users, the Swing link is the first link in the sidebar as shown in Figure. Q3.1 (b).

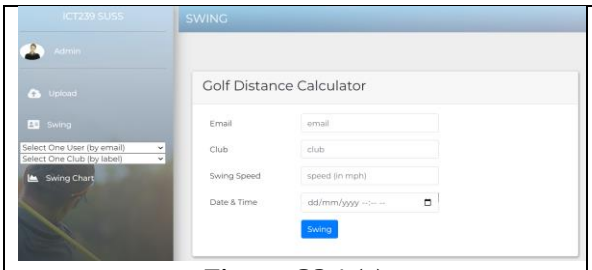


Figure. Q3.1 (a)

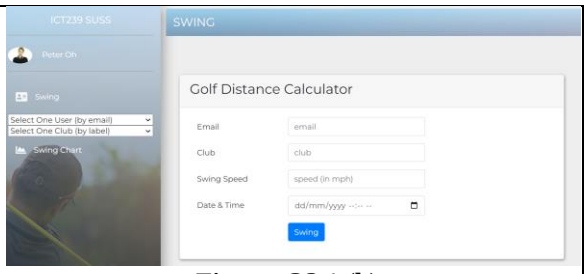


Figure. Q3.1 (b)

Notice that there are two dropdown lists below the Swing link.

The first dropdown list allows the user to select which golfer the swing will be recorded for in the database. The first dropdown list for admin user includes the emails of all registered users as shown in Figure Q3.2 (a) whereas for non-admin user, only the email of the logged in user is in the drop-down list as shown in Figure Q3.2 (a).

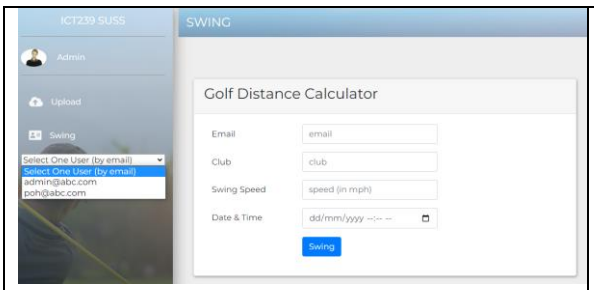


Figure. Q3.2 (a)

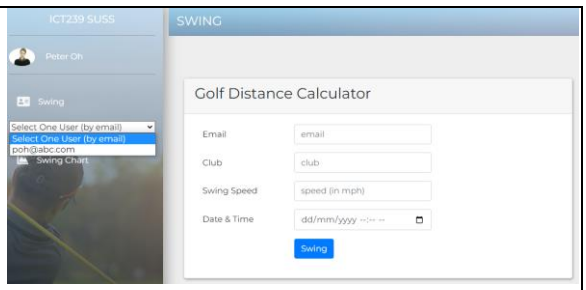


Figure. Q3.2 (b)

Note: From this point onwards, only the non-admin user's view will be shown in the Figureures as the interactivity for the admin user is the same as that for the non-admin user, except for the additional Upload link which is only available to admin user.

When a registered user is selected in the first dropdown list, the chosen email is reflected on the Golf Distance Calculator's email field; at the same time, the second dropdown list is populated with the clubs in the golf set for the chosen email as shown in Figure. Q3.3 (a). The population of the second dropdown list should be carried out without having to reload the html page.

The second dropdown list allows the user to select which club is used for the swing to be recorded in the database. When a club in the second dropdown list is selected, the club is reflected on the Golf Distance Calculator's club field as shown in the following Figure. Q3.3 (b). After values are entered into the Swing Speed and Date & Time fields, the Swing button is clicked to submit Swing data to the backend. The backend components calculate the distance and store the data/results into the database collections implemented in Question 2. The html page will then be reloaded to show either Figure. Q3.2 (a) or Figure. Q3.2 (b).

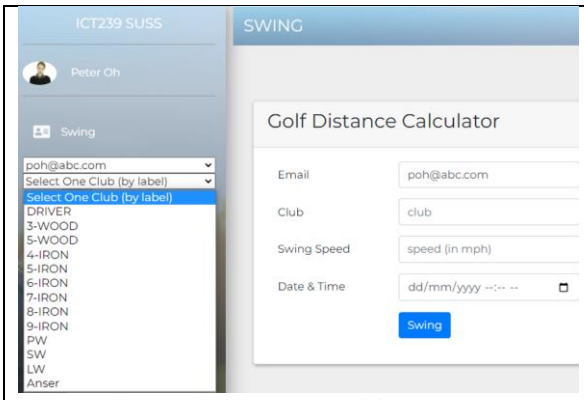


Figure. Q3.3 (a)

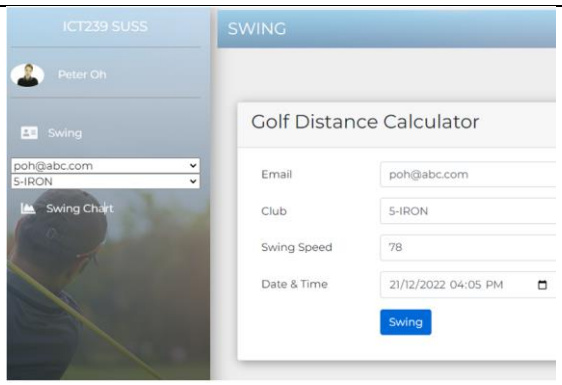


Figure. Q3.3 (b)

(a) Implement the frontend components for the Swing function for both admin and non admin users, including the components for interactivity. Explain what each component achieves for the Swing function.

(12 marks)

(b) Implement the backend components for the Swing function for both admin and non admin users. Explain what each component achieves for the Swing function.

Note: Put mongodb access methods/functions in the classes that define the relevant model(s).

(8 marks)

Reuse code where possible in your implementation, to avoid/reduce code duplication.

Question 4 (10 marks)

Learning objective: Demonstrate the visualisation of data on a web presentation

Implement the Swing Chart function such that when Swing Chart link in the sidebar is clicked, a chart will be shown with date-time as the X-axis and the distance swung as the Y-axis.

The number of line graphs in the chart depends on which user is logged in: if admin user is logged in, there is a line graph per golfer for all his swing date-time and distance as data points as shown in Figure. Q4.1 (a). If non-admin user is logged in, the chart will show a line graph if he has swings recorded as shown in Figure. Q4.1 (a).

(a) Implement the frontend components for the Swing Chart function for both admin and non admin users, including the components for interactivity. Explain what each component achieves for the Swing Chart function.

(4 marks)

(b) Implement the backend components of the Swing Chart function for both admin and non admin users. Explain what each component achieves for the Swing Chart function.

Note: Put mongodb access methods/functions in the classes that define the relevant model(s).

(6 marks)

Reuse code where possible in your implementation, to avoid/reduce code duplication.

Figure. Q4.1 (a)

Figure. Q4.1 (b)

---- END OF ASSIGNMENT ----