BIT216 Software Engineering Principles Assignment 3

Project HELPFit Fitness

Wong Wai Kiat B1301744
Chok Heng Nian B1500599

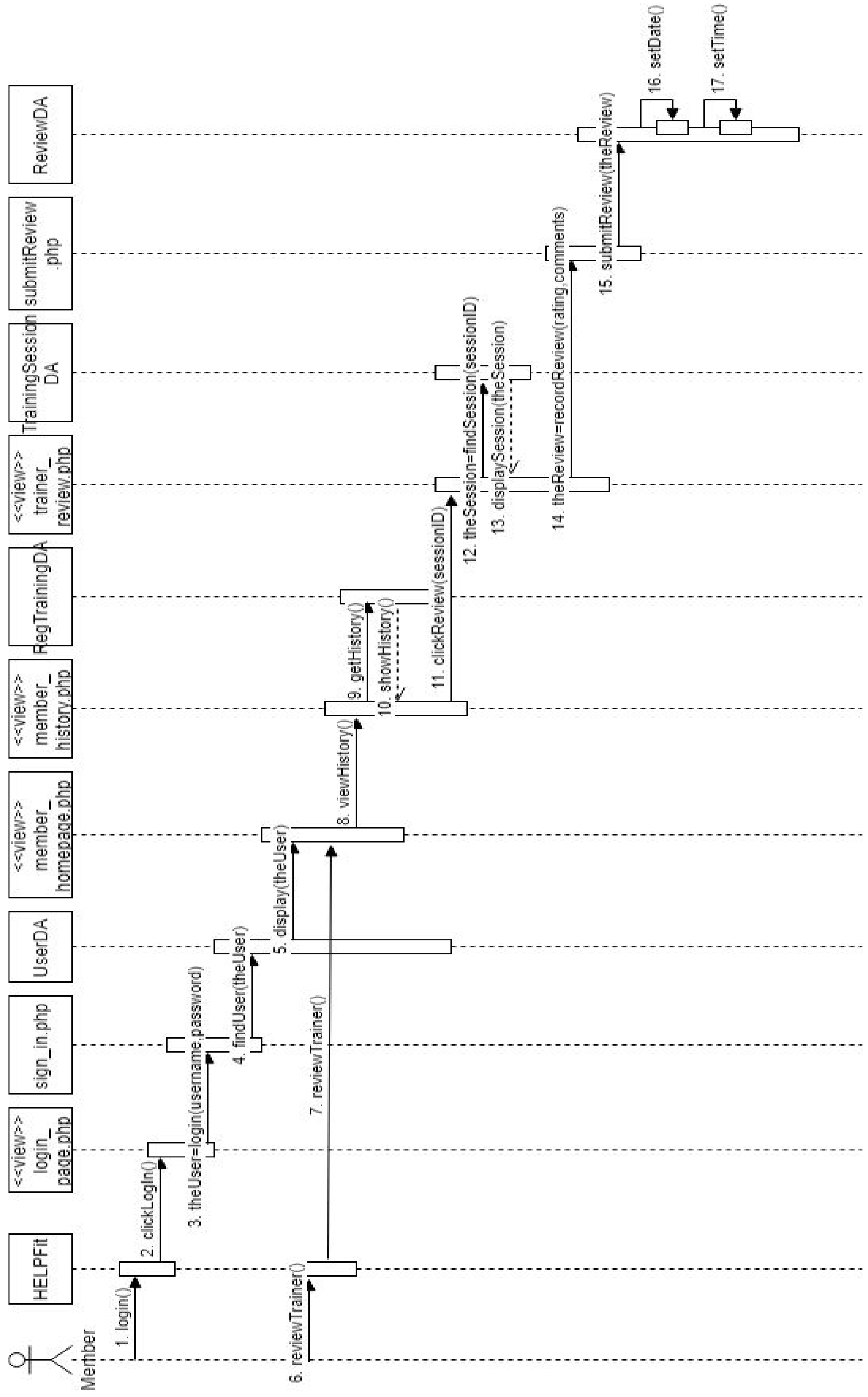
Table of Contents

1. Identification of Problems

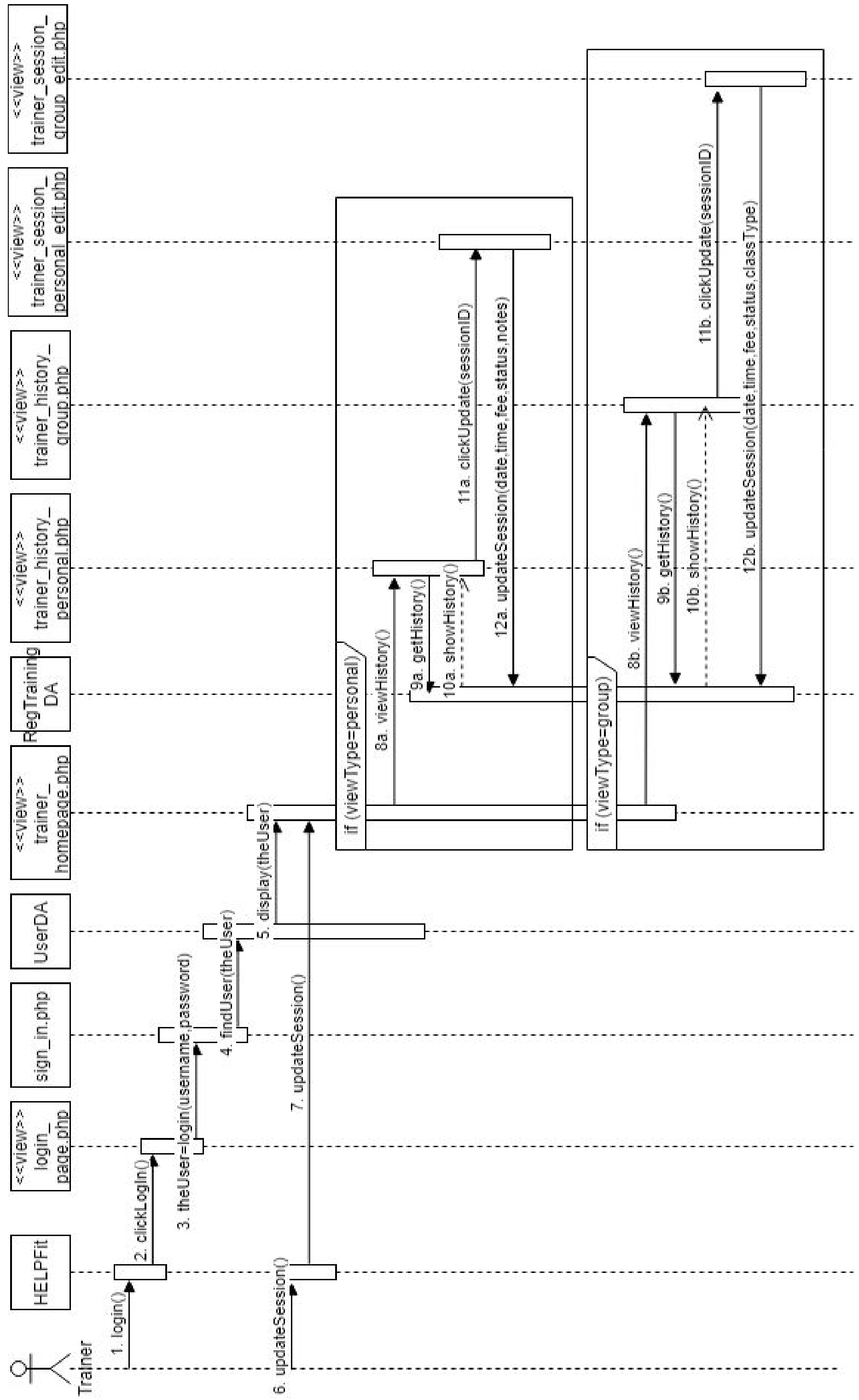
No.	Problems found in Iteration 1	Solution
1.	For the Register Personal or Group Training Session, in order to pass the session id value from the table using a button to send the value to the action php script page for inserting the id value into the database. It is difficult to be implemented and time consuming while working on this feature.	Firstly, to declare a class named “id”on sessionID column on the table, and using Javascript to to get the text value and send the value attached at end of the action php script page link and use Get method to get the id.
2.	To check whether the amount of registered members has reached the maximum amount of participants in the database for the Register group training session. My teammates and I were confused which section in the coding should implement this feature and it is time consuming.	To check which section of the code at the register group training session page is executing the sql query to insert the data into the database and then increment the count every time when the member has registered on the specific group training session and implement the condition to check whether it has reached the amount of maximum participants if the condition matched update the status to “FULL” or update the count in the database

2. Analysis/Design Updates

Sequence Diagram (Use Case 5: Review Trainer)



Sequence Diagram (Use Case 6: Update Training Session)


Trainer
HELPFit
<<view>> login_ page.php
sign_in.php
UserDA
<<view>> trainer_ homepage.php
RegTraining DA
<<view>> trainer_history_ personal.php
<<view>> trainer_history_ group.php
<<view>> trainer_session_ personal_edit.php
<<view>> trainer_session_ group_edit.php
1. login()
2. clickLogin()
3. theUser=login(username,password)
4. findUser(theUser)
5. display(theUser)
6. updateSession()
7. updateSession()
if (viewType=personal)
8a. viewHistory()
9a. getHistory()
10a. showHistory()
11a. clickUpdate(sessionID)
12a. updateSession(date,time,fee,status,notes)
if (viewType=group)
8b. viewHistory()
9b. getHistory()
10b. showHistory()
11b. clickUpdate(sessionID)
12b. updateSession(date,time,fee,status,classType)

3. Test Objectives

- To check for errors in the looks of the user interface.
- To discover errors in algorithms design for each use case.
- To refine the integration within the system.
- To ensure Prototype 2 meets all the user requirements.

4. Updated Test Plan

Type of testing	Approach	Use Case	Schedule
Unit testing	White box	Review Trainer	26/10/17 - 04/11/17
Unit testing	White box	Update Training Session	26/10/17 - 04/11/17
Integration testing	Black box, top-down	Review Trainer	5/11/17 – 12/11/17
Integration testing	Black box, top-down	Update Training Session	5/11/17 – 12/11/17
System testing (Functional)	Black box	Review Trainer	5/11/17 – 12/11/17
System testing (Functional)	Black box	Update Training Session	5/11/17 – 12/11/17
System testing (Non-functional)	Black box	Review Trainer	5/11/17 – 12/11/17
System testing (Non-functional)	Black box	Update Training Session	5/11/17 – 12/11/17

5. Test methods specification evaluation

- Google Chrome web browser is used for most of the test cases.
- Other web browsers such as Internet Explorer, Mozilla Firefox, and Safari are also used.
- Atom is used to check on the source codes for unit testing.
- XAMPP is used to test the connection of the system to the database server. Database is checked consistently for each test case.
- GanttProject is used to schedule and plan the whole testing process.

6. Unit Testing

Unit testing for Use Case 5: Review Trainer

Test Case 1: Member review validation

Input

Clicking on ‘Review’ button:

a) When training session is still going on.
b) When training session is completed.

Source code

```
<?php
  if(!($passData)){
  if($rs['status'] == 'Completed'){
?>
  <p><a class="btn btn-primary" href="trainer_review.php?session_id=<?php
      echo $rs['sessionID']; ?>" role="button">Review &raquo;</a></p>
<?php }
   else {
?>
      <p><a class="btn btn-primary disabled"
      href="trainer_review.php?session_id=<?php echo $rs['sessionID']; ?>"
      role="button">Review &raquo;</a></p>
<?php
    }}else{
?>
    <p><a class="btn btn-primary disabled"
     href="trainer_review.php?session_id=<?php echo $rs['sessionID']; ?>"
     role="button">Review &raquo;</a></p>
 <?php
   }
 ?>
</div>
```

Expected outcome

The specific button will be disabled in a) and it will be enabled in b).

Actual outcome

a) ‘Review’ button is disabled.
b) ‘Review’ button is enabled.

Test Case 2: Trainer check review validation

Input

Clicking on ‘Check Reviews’ button:

a) When training session is still going on.
b) When training session is completed.

Source code

```
<?php
 $check2 ="Completed";
 if($rs["status"] == $check2){
?>
   <td><button class="btn btn-info" data-toggle="modal" data-target="#test"
   data-id="<?php echo $session_id;?>">Check Review</button></td>
<?php } else{?>
   <td><button class="btn btn-info disabled">Check Review</button></td>
<?php
   }
```

Expected outcome

The specific button will be disabled in a) and it will be enabled in b).

Actual outcome

a) ‘Check Reviews’ button is disabled.
b) ‘Check Reviews’ button is enabled.

Unit testing for Use Case 6: Update Training Session

Test Case 1: Trainer update session validation

Input

Clicking on ‘Edit’ button:

a) When training session is still going on.
b) When training session is completed.

Source code

```
<?php
 if($rs["status"] !== $check2){
?>
  <td>
    <div class="text-center">
      <button type="submit" name="submit" class="btn btn-default
          btnEditPer"><em class="fa fa-pencil"></em></button>
     </div>
   </td>
<?php } else{?>
   <td>
      <div class="text-center">
         <button class="btn btn-default  disabled"><em class="fa fa-pencil"></em></button>
       </div>
    </td>
 <?php
   }
 ?>
```

Expected outcome

The specific button in a) will be disabled and it will be enabled in b)

Actual outcome

a) ‘Edit’ button is disabled.
b) ‘Edit’ button is enabled.

Test Case 2: Train update session - ‘Fee’ value input

Input

Input (a) Invalid ‘Fee’ value

Data	**Value**
Title	Personal 1
Date	20 November 2017
Time	11:11:11 AM
Fee	One hundred and ten
Notes	null
Status	Available

Input (b) Valid ‘Fee’ value

Data	**Value**
Title	Personal 1
Date	20 November 2017
Time	11:11:11 AM
Fee	110
Notes	null
Status	Available

Source code

```
if(is_numeric($fee)){
                    ……....
                    ……….
}else{
          echo '<script language = "javascript">';
          echo 'alert("Fee must be in numeric format!")';
          echo '</script>';
          echo  "<script
          window.location.assign('trainer_session_personal_edit.php?id=$session_id');
           </script>";
          $sessionID = $_SESSION['PrSessionUpdate'];
}
```

Expected outcome

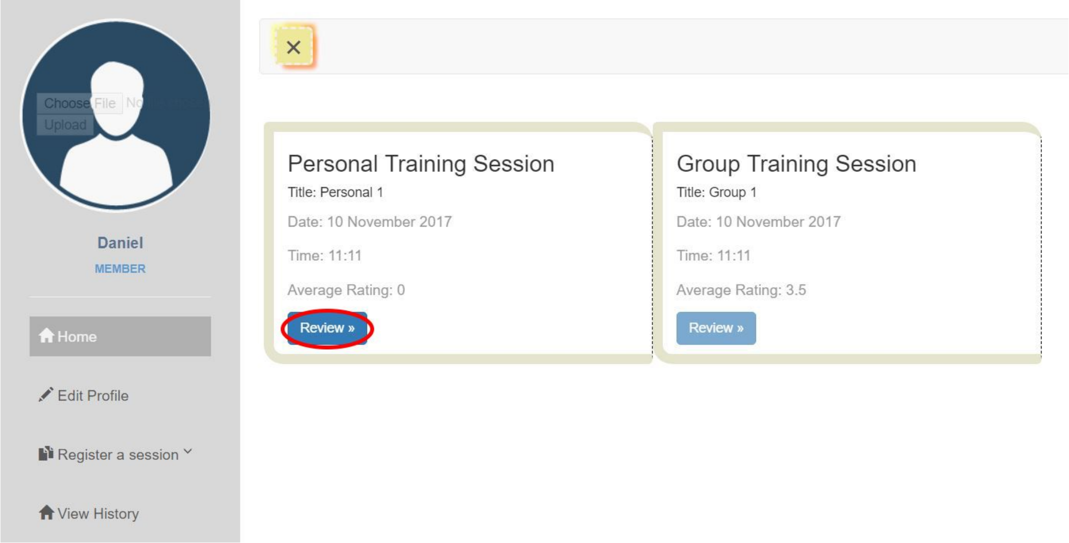
a) An error message will be displayed.
b) A successful message will be displayed.

Actual outcome

a) “Fee must be in numeric format!” dialog box pop-up.
b) “Record Updated successfully!” dialog box pop-up.

7. Integration testing

Integration test for Use Case 5: Review Trainer

Test Case 1: Navigate to review page
Input


Personal Training Session

Title: Personal 1

Date: 10 November 2017

Time: 11:11

Average Rating: 0

Group Training Session

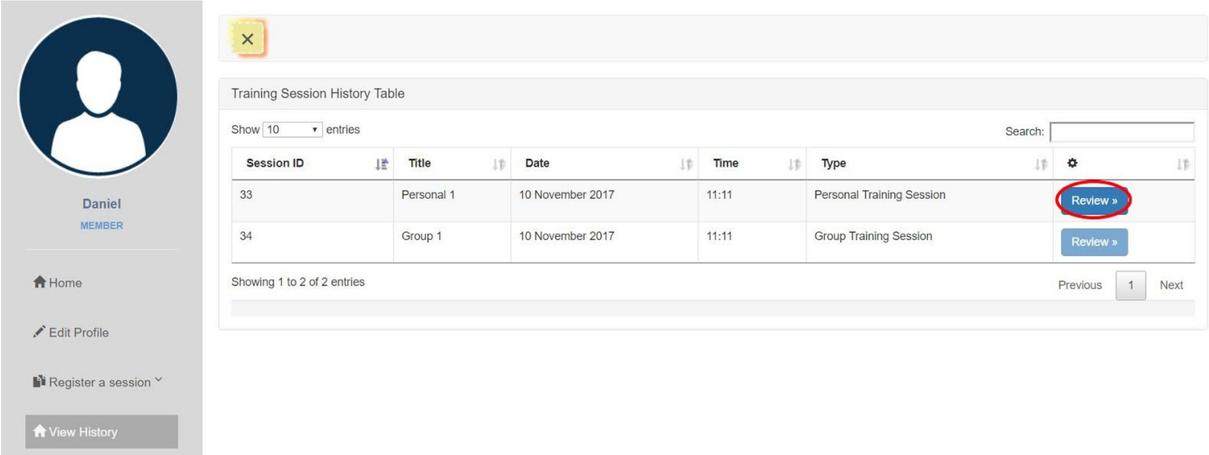
Title: Group 1

Date: 10 November 2017

Time: 11:11

Average Rating: 3.5

Option (a): Clicking "Review" button on Member Homepage



Training Session History Table

Session ID	Title	Date	Time	Type	⚙
33	Personal 1	10 November 2017	11:11	Personal Training Session	Review »
34	Group 1	10 November 2017	11:11	Group Training Session	Review »

Showing 1 to 2 of 2 entries

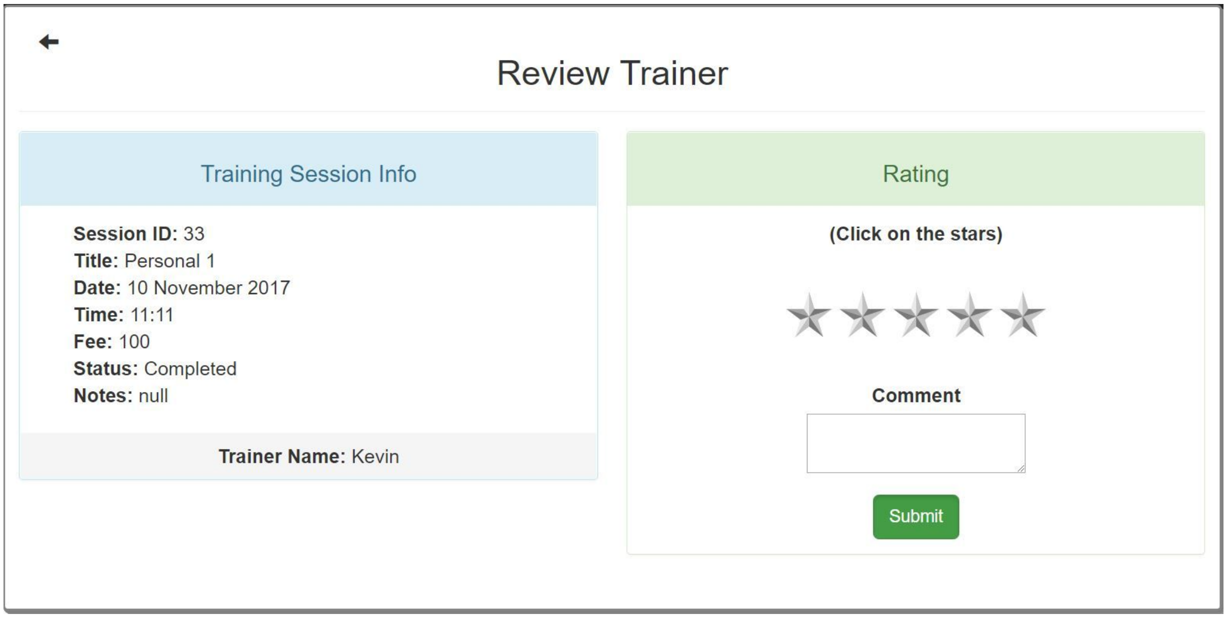
Option (b): Clicking "Review" button on Training Session History table

Expected outcome

The member will be navigated from either:

a) Member Homepage to Review Trainer page.
b) Training Session History to Review Trainer page.

Actual outcome



Review Trainer

Training Session Info

Session ID: 33
Title: Personal 1
Date: 10 November 2017
Time: 11:11
Fee: 100
Status: Completed
Notes: null

Trainer Name: Kevin

Rating

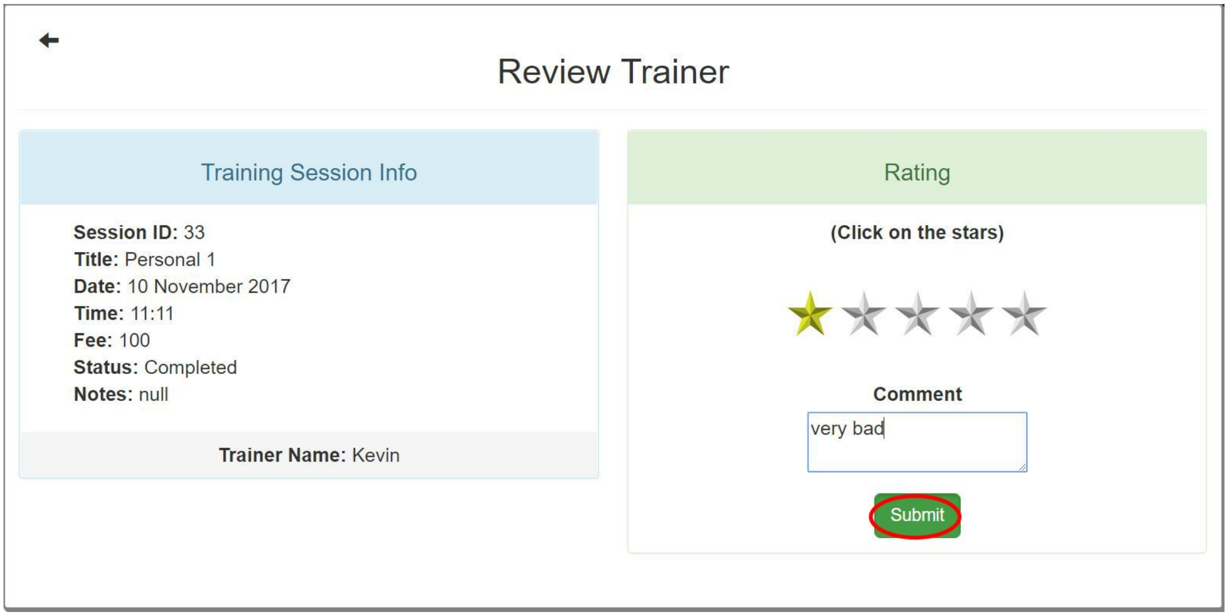
(Click on the stars)

Comment

Review Trainer page

Test Case 2: Submit review

Input



Review Trainer

Training Session Info

Session ID: 33
Title: Personal 1
Date: 10 November 2017
Time: 11:11
Fee: 100
Status: Completed
Notes: null

Trainer Name: Kevin

Rating

(Click on the stars)

★☆☆☆☆

Comment

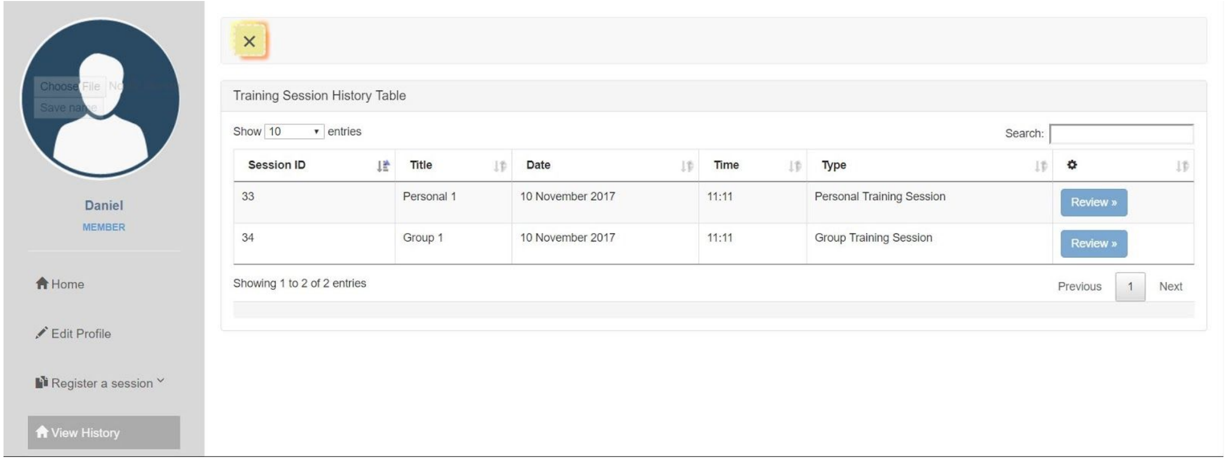
very bad

Clicking ‘Submit’ button after filling in rating and comments

Expected outcome

The member will be navigated from Review Trainer page back to Training Session History page.

Actual outcome



Training Session History Table

Show 10 entries

Search:

Session ID	Title	Date	Time	Type	⚙
33	Personal 1	10 November 2017	11:11	Personal Training Session	Review »
34	Group 1	10 November 2017	11:11	Group Training Session	Review »

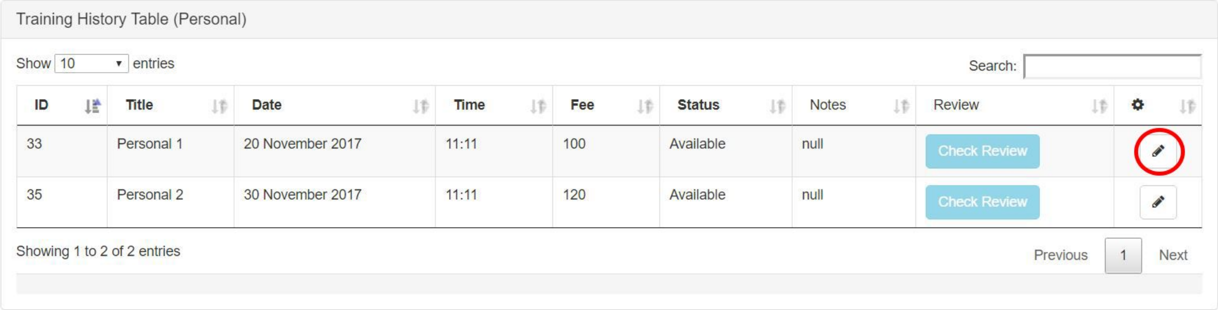
Showing 1 to 2 of 2 entries

Training Session History page

Integration testing for Use Case 6: Update Training Session

Test Case 1: Navigate to Update Training Session page

Input



Training History Table (Personal)

Show 10 entries

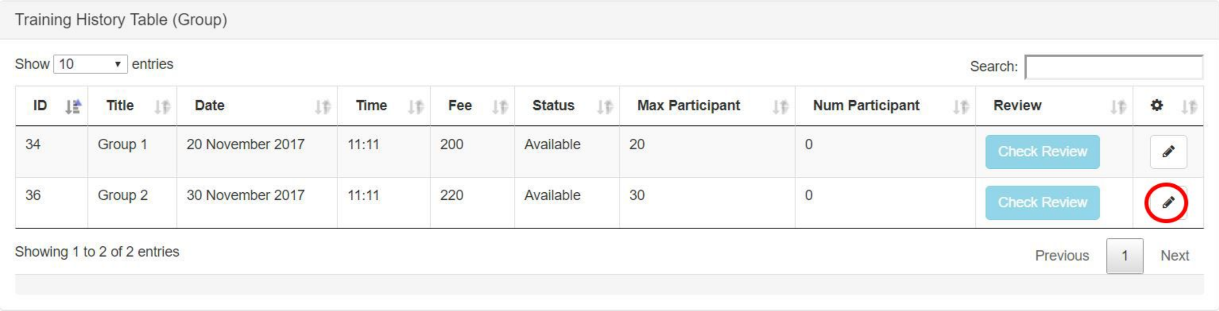
Search:

ID	Title	Date	Time	Fee	Status	Notes	Review	⚙
33	Personal 1	20 November 2017	11:11	100	Available	null	Check Review	✎
35	Personal 2	30 November 2017	11:11	120	Available	null	Check Review	✎

Showing 1 to 2 of 2 entries

Previous 1 Next

a) Clicking ✎ button on Training History Table (Personal)



Training History Table (Group)

Show 10 entries

Search:

ID	Title	Date	Time	Fee	Status	Max Participant	Num Participant	Review	⚙
34	Group 1	20 November 2017	11:11	200	Available	20	0	Check Review	✎
36	Group 2	30 November 2017	11:11	220	Available	30	0	Check Review	✎

Showing 1 to 2 of 2 entries

Previous 1 Next

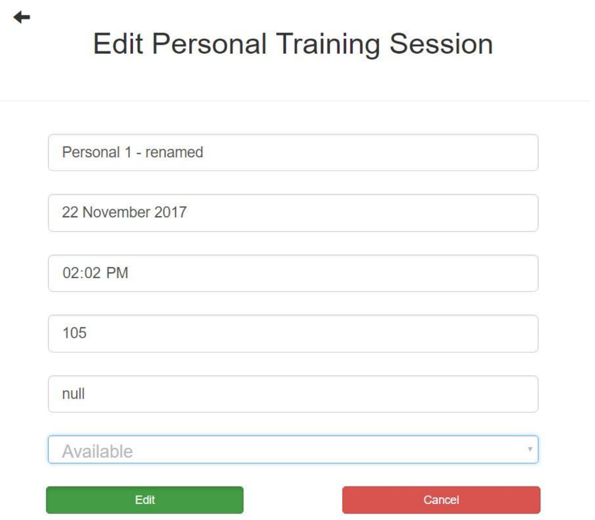
b) Clicking ✎ button on Training History Table (Group)

Expected outcome

The trainer will be navigated from:

a) Trainer History Personal page to Edit Personal Training Session page.
b) Trainer History Group page to Edit Group Training Session page.

Actual outcome



Edit Personal Training Session

Personal 1 - renamed

22 November 2017

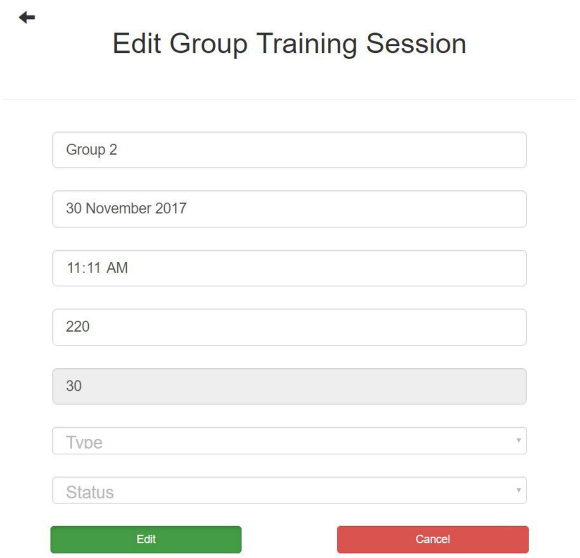
02:02 PM

105

null

Available

a) Edit Personal Training Session page



Edit Group Training Session

Group 2

30 November 2017

11:11 AM

220

30

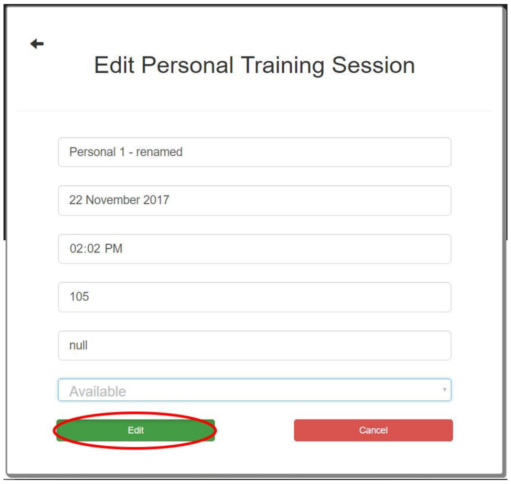
Type

Status

b) Edit Group Training Session page

Test Case 2: Submit updated training session details

Input



Edit Personal Training Session

Personal 1 - renamed

22 November 2017

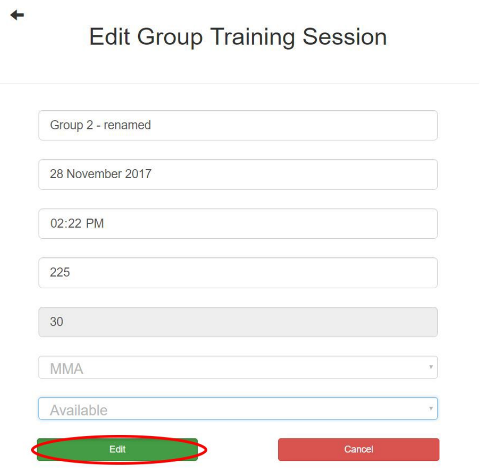
02:02 PM

105

null

Available

a) Clicking ‘Edit’ button on Edit Personal Training Session page



Edit Group Training Session

Group 2 - renamed

28 November 2017

02:22 PM

225

30

MMA

Available

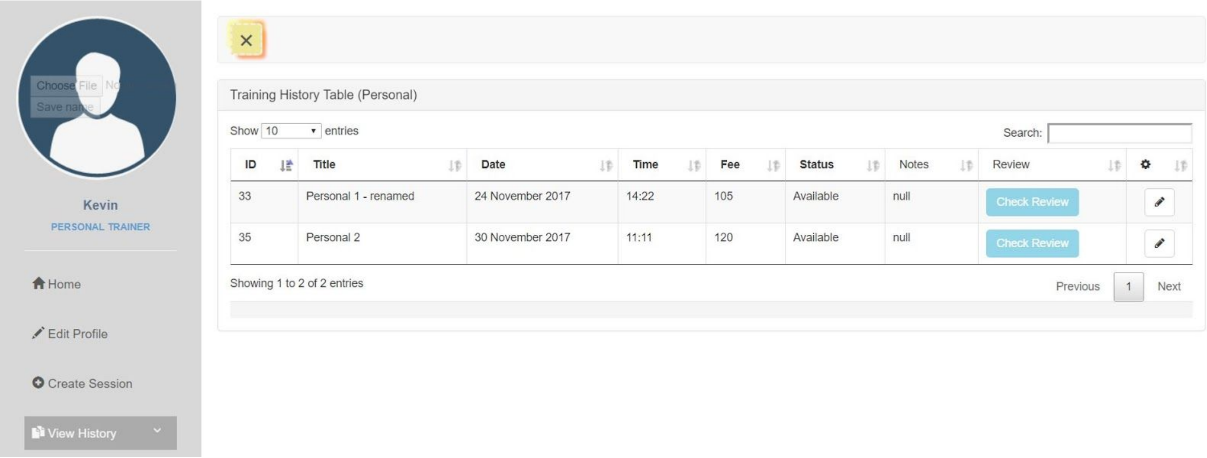
b) Clicking ‘Edit’ button on Edit Group Training Session page

Expected outcome

Trainer will be redirected either:

a) From Edit Personal Training Session page to Training History Personal page.
b) From Edit Group Training Session page to Training History Group page.

Actual outcome

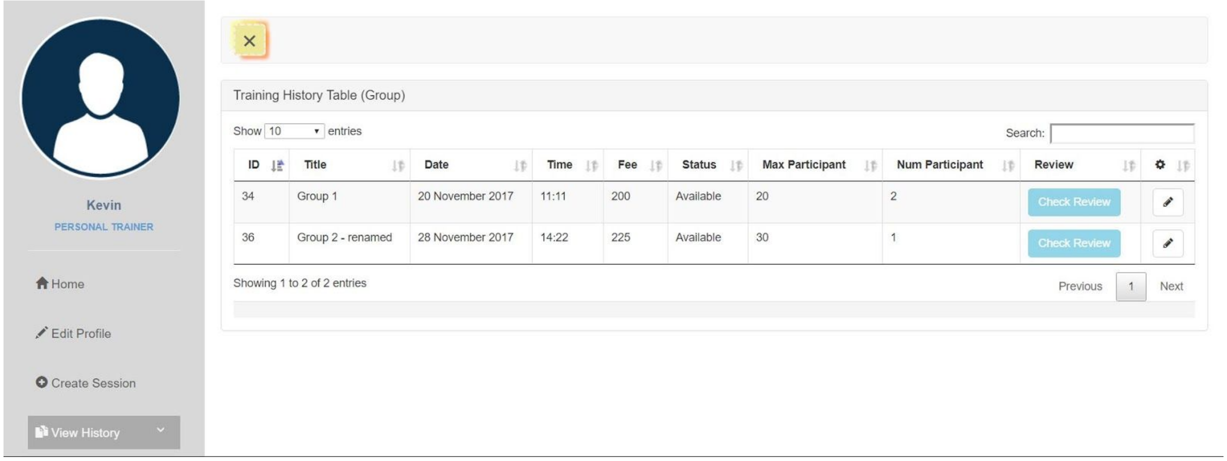


Training History Table (Personal)

ID	Title	Date	Time	Fee	Status	Notes	Review	⚙
33	Personal 1 - renamed	24 November 2017	14:22	105	Available	null	Check Review	✎
35	Personal 2	30 November 2017	11:11	120	Available	null	Check Review	✎

Showing 1 to 2 of 2 entries

a) Training History Personal page



Training History Table (Group)

ID	Title	Date	Time	Fee	Status	Max Participant	Num Participant	Review	⚙
34	Group 1	20 November 2017	11:11	200	Available	20	2	Check Review	✎
36	Group 2 - renamed	28 November 2017	14:22	225	Available	30	1	Check Review	✎

Showing 1 to 2 of 2 entries

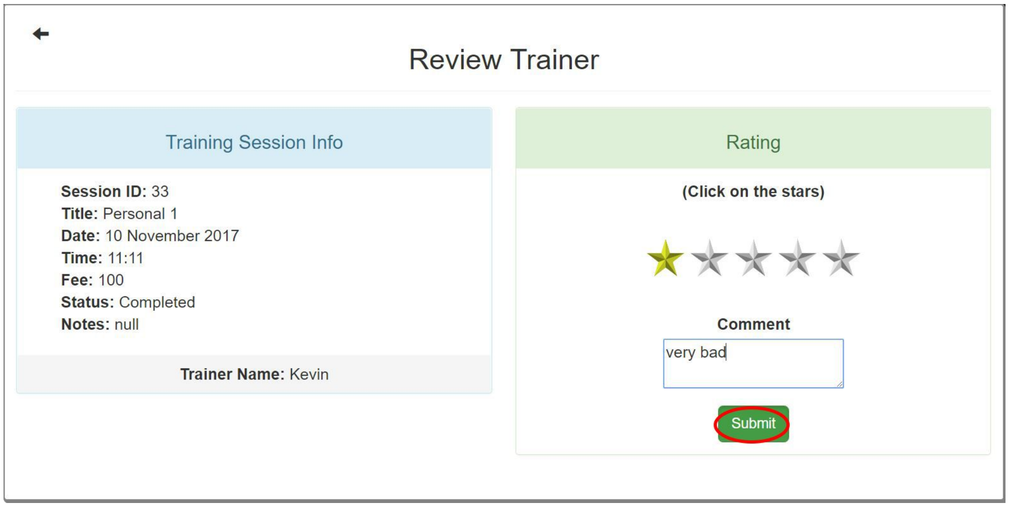
b) Training History Group page

8. System testing (Functional)

System testing for Use Case 5: Review Trainer

Test Case 1: Review Trainer (Personal)

Input



Review Trainer

Training Session Info

Session ID: 33
Title: Personal 1
Date: 10 November 2017
Time: 11:11
Fee: 100
Status: Completed
Notes: null

Trainer Name: Kevin

Rating

(Click on the stars)

★☆☆☆☆

Comment

very bad

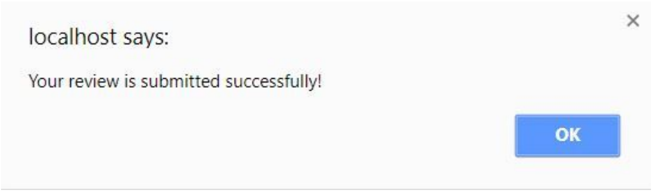
Submit

Member ‘Daniel’ reviewing session ‘Personal 1’

Expected outcome

Pop up dialog “Your review is submitted successfully!” will be displayed.
All inputs will be recorded and stored in database table ‘review’.
Current date and timestamp will be recorded together with the review.

Actual outcome

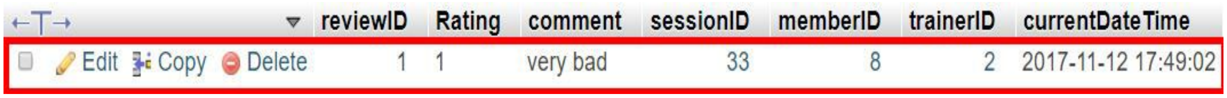


localhost says:

Your review is submitted successfully!

OK

Dialog box

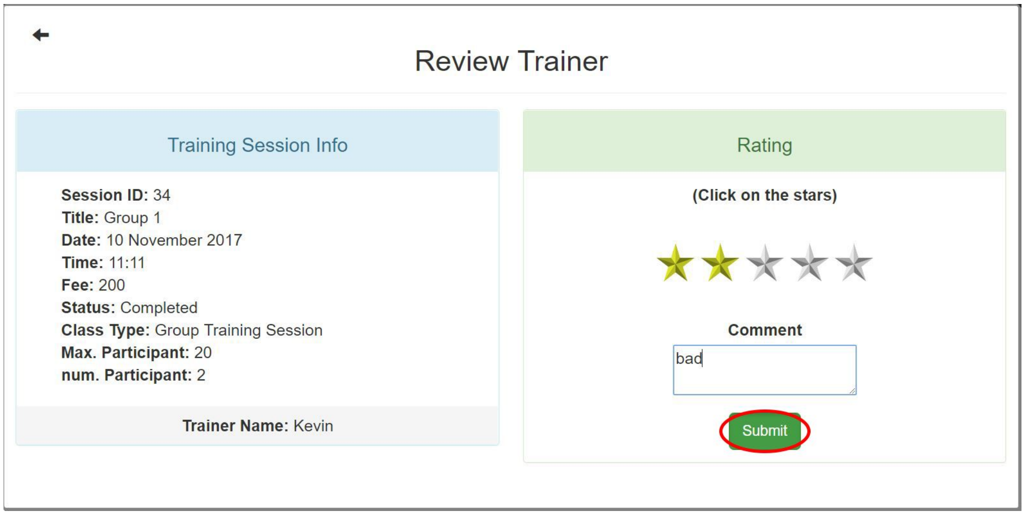


←T→	reviewID	Rating	comment	sessionID	memberID	trainerID	currentDateTime
□ Edit Copy Delete	1	1	very bad	33	8	2	2017-11-12 17:49:02

‘review’ table in database

Test Case 2: Review Trainer (Group)

Input



Review Trainer

Training Session Info

Session ID: 34
Title: Group 1
Date: 10 November 2017
Time: 11:11
Fee: 200
Status: Completed
Class Type: Group Training Session
Max. Participant: 20
num. Participant: 2

Trainer Name: Kevin

Rating

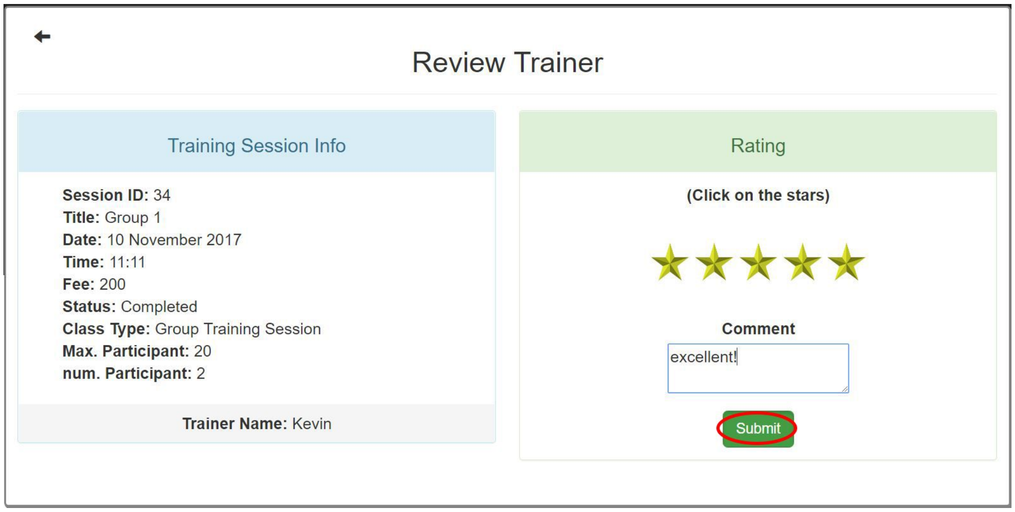
(Click on the stars)

★★☆☆☆

Comment

bad

First review of session ‘Group 1’ by Member ‘Daniel’



Review Trainer

Training Session Info

Session ID: 34
Title: Group 1
Date: 10 November 2017
Time: 11:11
Fee: 200
Status: Completed
Class Type: Group Training Session
Max. Participant: 20
num. Participant: 2

Trainer Name: Kevin

Rating

(Click on the stars)

★★★★★

Comment

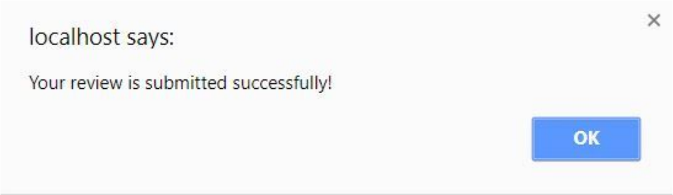
excellent!

Second review of session ‘Group 1’ by Member ‘Himu’

Expected outcome

Pop up dialog “Your review is submitted successfully!” will be displayed.
All inputs will be recorded and stored in database table ‘review’.
Current date and timestamp will be recorded together with the review.

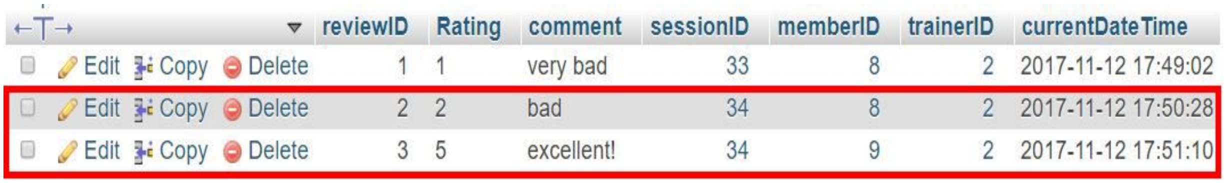
Actual outcome



localhost says:

Your review is submitted successfully!

Dialog box

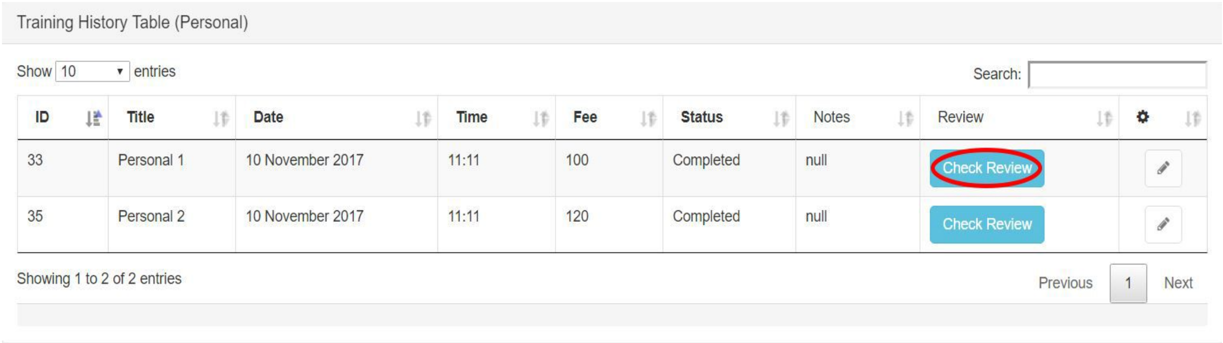


←T→	reviewID	Rating	comment	sessionID	memberID	trainerID	currentDateTime
□ Edit Copy Delete	1	1	very bad	33	8	2	2017-11-12 17:49:02
□ Edit Copy Delete	2	2	bad	34	8	2	2017-11-12 17:50:28
□ Edit Copy Delete	3	5	excellent!	34	9	2	2017-11-12 17:51:10

Updated ‘review’ table in database

Test Case 3: Trainer Check Reviews (Personal)

Input



Training History Table (Personal)

Show 10 entries Search:

ID	Title	Date	Time	Fee	Status	Notes	Review	⚙
33	Personal 1	10 November 2017	11:11	100	Completed	null	Check Review	✎
35	Personal 2	10 November 2017	11:11	120	Completed	null	Check Review	✎

Showing 1 to 2 of 2 entries

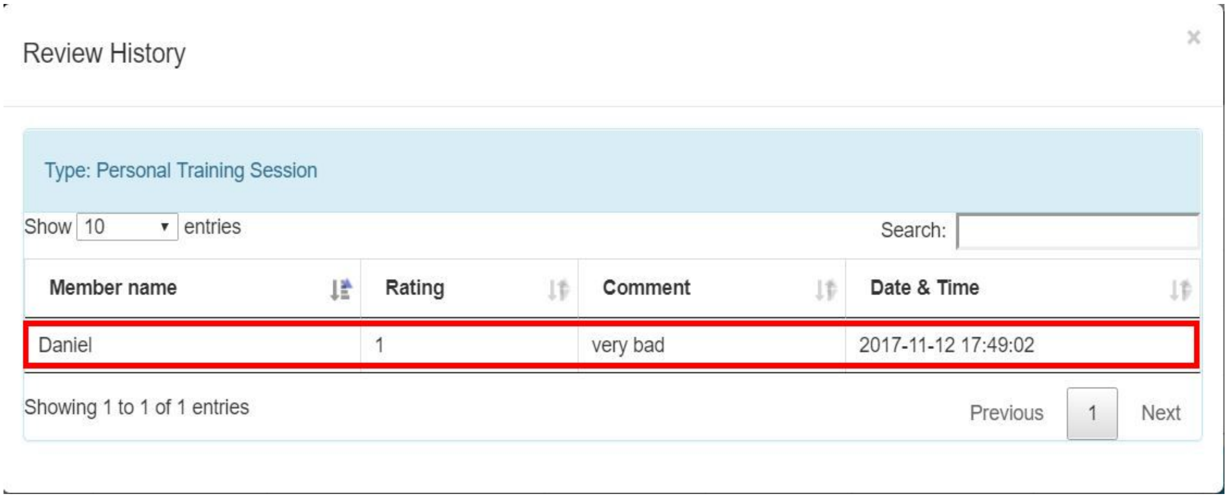
Previous 1 Next

Clicking on "Check Review" button for session 'Personal 1'

Expected outcome

Pop-up table showing review history for session 'Personal 1' will be displayed.

Actual outcome



Review History

Type: Personal Training Session

Show 10 entries Search:

Member name	Rating	Comment	Date & Time
Daniel	1	very bad	2017-11-12 17:49:02

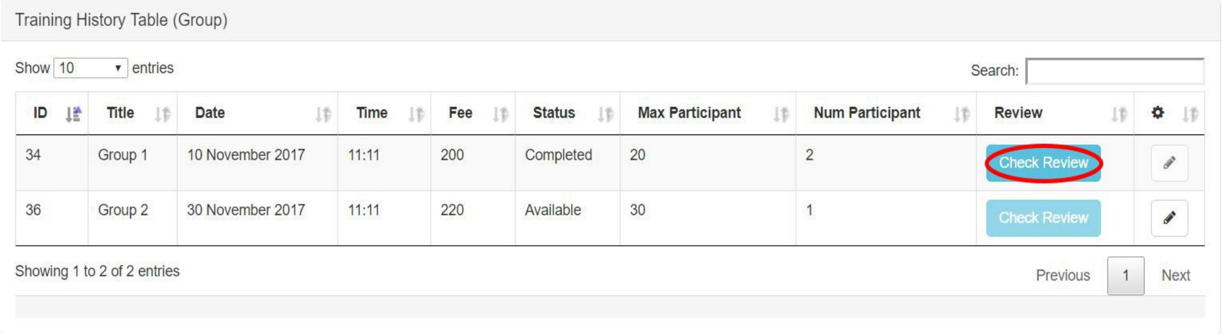
Showing 1 to 1 of 1 entries

Previous 1 Next

Review History table for session 'Personal 1'

Test Case 4: Trainer Check Reviews (Group)

Input



Training History Table (Group)

Show 10 entries

Search:

ID	Title	Date	Time	Fee	Status	Max Participant	Num Participant	Review	⚙
34	Group 1	10 November 2017	11:11	200	Completed	20	2	Check Review	✎
36	Group 2	30 November 2017	11:11	220	Available	30	1	Check Review	✎

Showing 1 to 2 of 2 entries

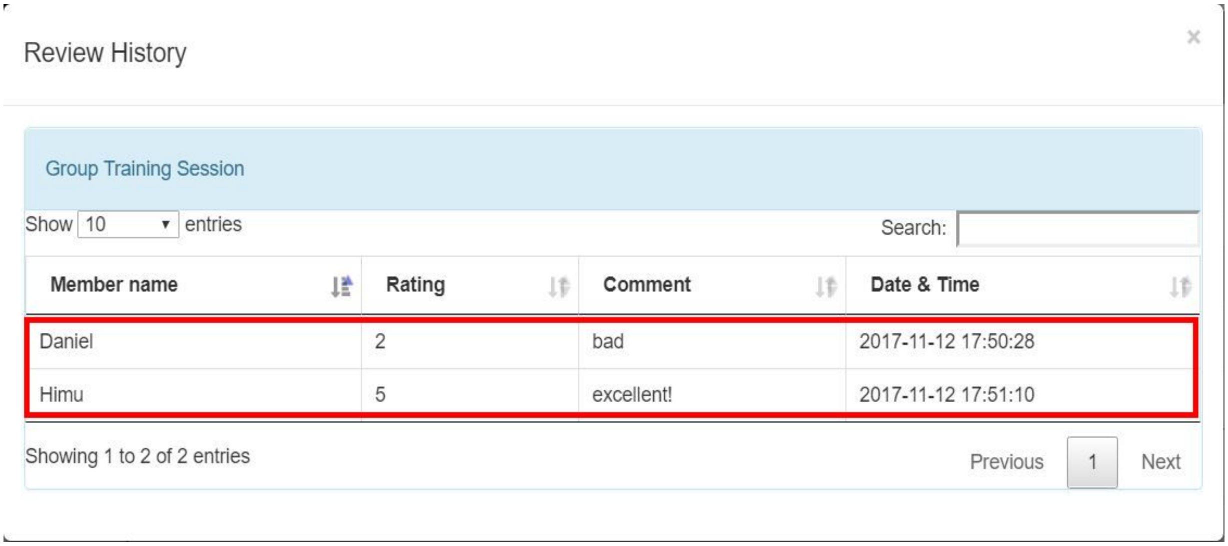
Previous 1 Next

Clicking on “Check Review” button for session ‘Group 1’

Expected outcome

Pop-up table showing review history for session ‘Group 1’ will be displayed.

Actual outcome



Review History

×

Group Training Session

Show 10 entries

Search:

Member name	Rating	Comment	Date & Time
Daniel	2	bad	2017-11-12 17:50:28
Himu	5	excellent!	2017-11-12 17:51:10

Showing 1 to 2 of 2 entries

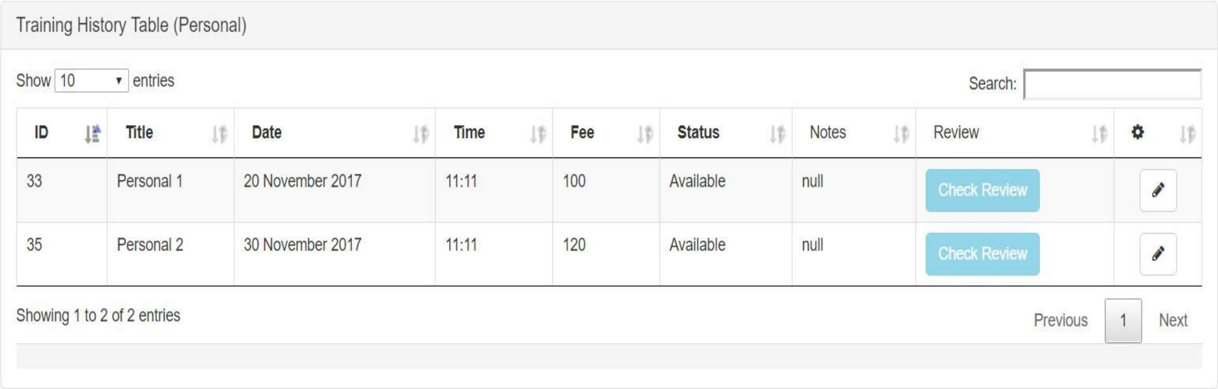
Previous 1 Next

Review History table for session ‘Group 1’

System testing for Use Case 6: Update Training Session

Test Case 1: Update Training Session (Personal)

Input



Training History Table (Personal)

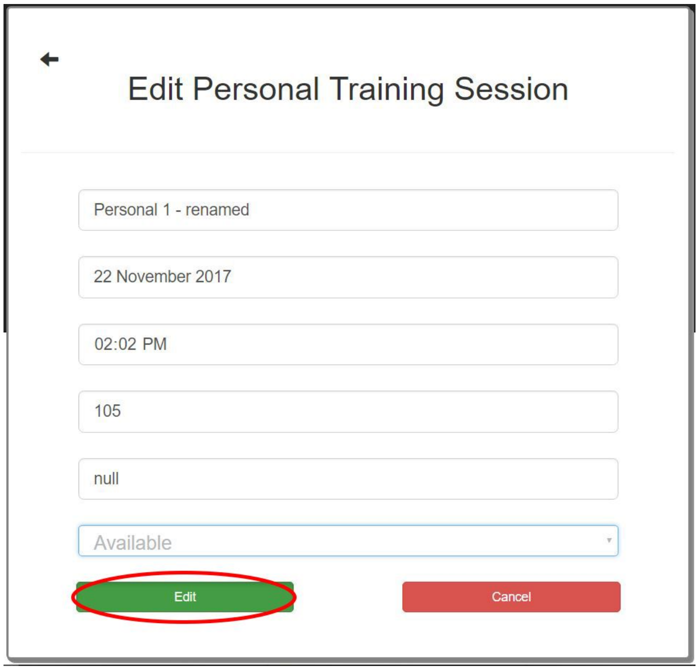
Show 10 entries

Search:

ID	Title	Date	Time	Fee	Status	Notes	Review	⚙
33	Personal 1	20 November 2017	11:11	100	Available	null	Check Review	✎
35	Personal 2	30 November 2017	11:11	120	Available	null	Check Review	✎

Showing 1 to 2 of 2 entries

Training History Table (Personal) before updating



Edit Personal Training Session

Personal 1 - renamed

22 November 2017

02:02 PM

105

null

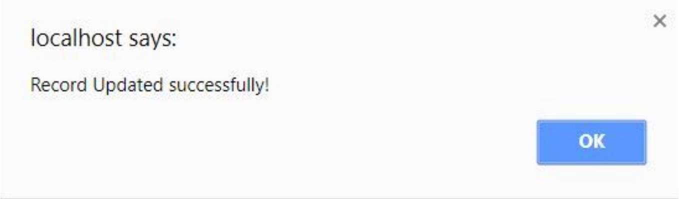
Available

Edit Personal Training Session form with new session details

Expected outcome

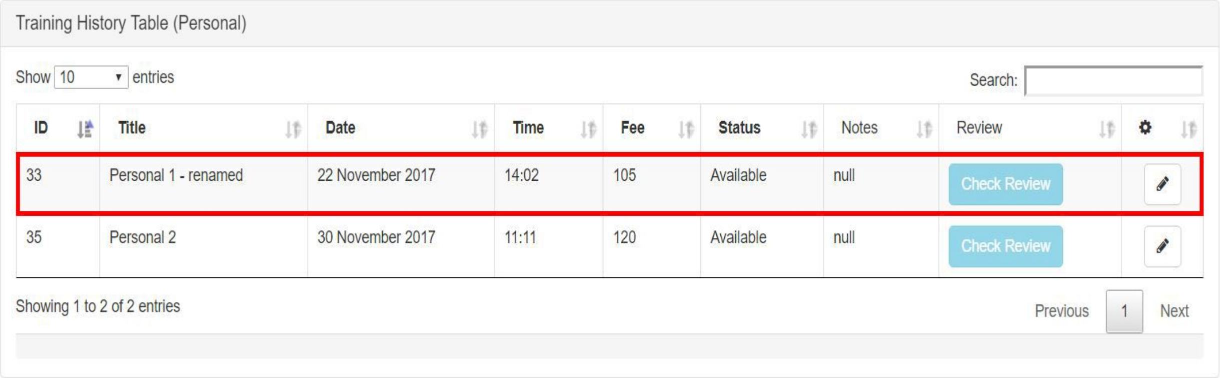
Pop up dialog “Record Updated successfully!” will be displayed.
All inputs will be recorded and updated in database table ‘review’.
Training Session History (Personal) table will be updated.

Actual outcome


localhost says:

Record Updated successfully!

Dialog box


Training History Table (Personal)

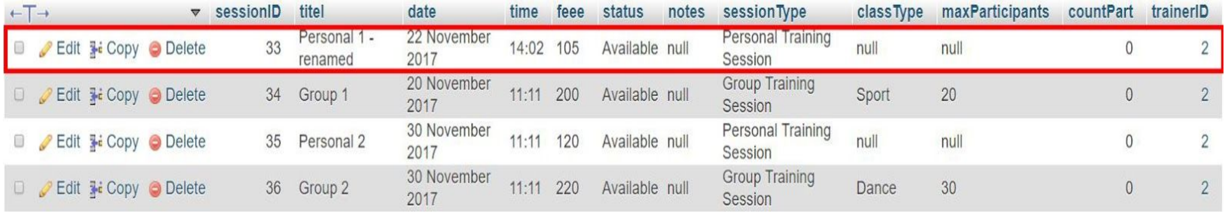
Show 10 entries

Search:

ID	Title	Date	Time	Fee	Status	Notes	Review	⚙
33	Personal 1 - renamed	22 November 2017	14:02	105	Available	null	Check Review	✎
35	Personal 2	30 November 2017	11:11	120	Available	null	Check Review	✎

Showing 1 to 2 of 2 entries

Updated Training History Table (Personal)

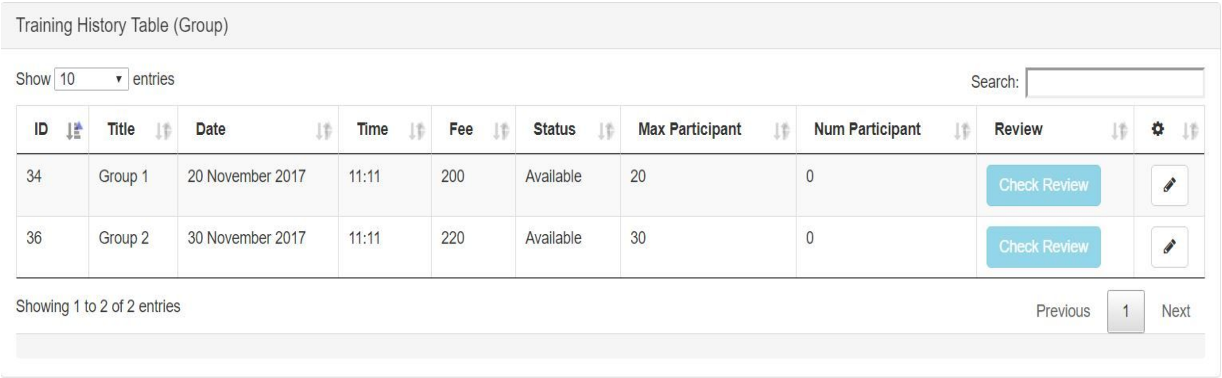


←T→	sessionID	titel	date	time	feee	status	notes	sessionType	classType	maxParticipants	countPart	trainerID
□ Edit Copy Delete	33	Personal 1 - renamed	22 November 2017	14:02	105	Available	null	Personal Training Session	null	null	0	2
□ Edit Copy Delete	34	Group 1	20 November 2017	11:11	200	Available	null	Group Training Session	Sport	20	0	2
□ Edit Copy Delete	35	Personal 2	30 November 2017	11:11	120	Available	null	Personal Training Session	null	null	0	2
□ Edit Copy Delete	36	Group 2	30 November 2017	11:11	220	Available	null	Group Training Session	Dance	30	0	2

Updated ‘review’ table in database

Test Case 2: Update Training Session (Group)

Input



Training History Table (Group)

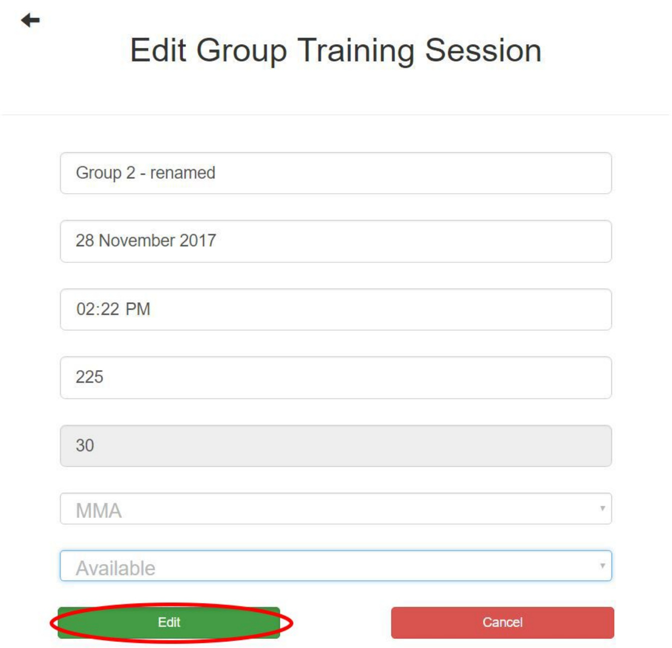
Show 10 entries

Search:

ID	Title	Date	Time	Fee	Status	Max Participant	Num Participant	Review	⚙
34	Group 1	20 November 2017	11:11	200	Available	20	0	Check Review	✎
36	Group 2	30 November 2017	11:11	220	Available	30	0	Check Review	✎

Showing 1 to 2 of 2 entries

Training History Table (Group) before updating



Edit Group Training Session

Group 2 - renamed

28 November 2017

02:22 PM

225

30

MMA

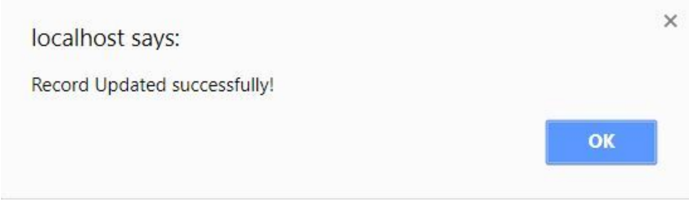
Available

Edit Group Training Session form with new session details

Expected outcome

Pop up dialog “Record updated successfully” will be displayed.
All inputs will be recorded and updated in database table ‘review’.
Training Session History (Group) table will be updated.

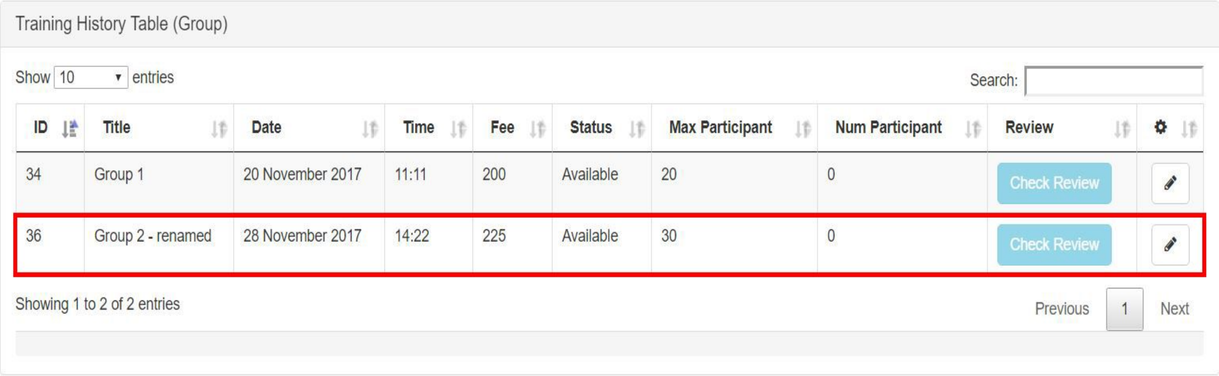
Actual outcome



localhost says:

Record Updated successfully!

Dialog box



Training History Table (Group)

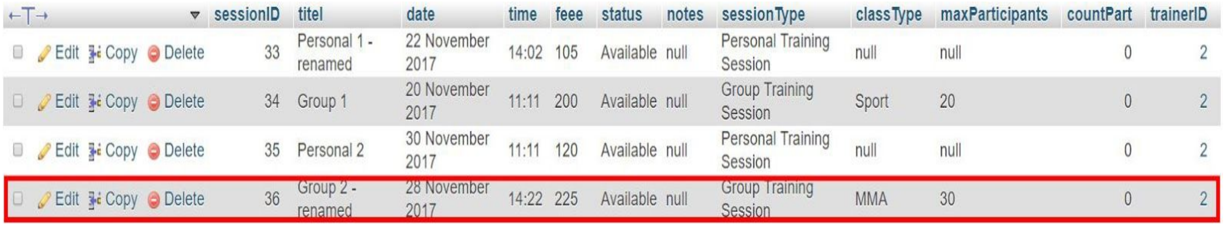
Show 10 entries

Search:

ID	Title	Date	Time	Fee	Status	Max Participant	Num Participant	Review	⚙
34	Group 1	20 November 2017	11:11	200	Available	20	0	Check Review	✎
36	Group 2 - renamed	28 November 2017	14:22	225	Available	30	0	Check Review	✎

Showing 1 to 2 of 2 entries

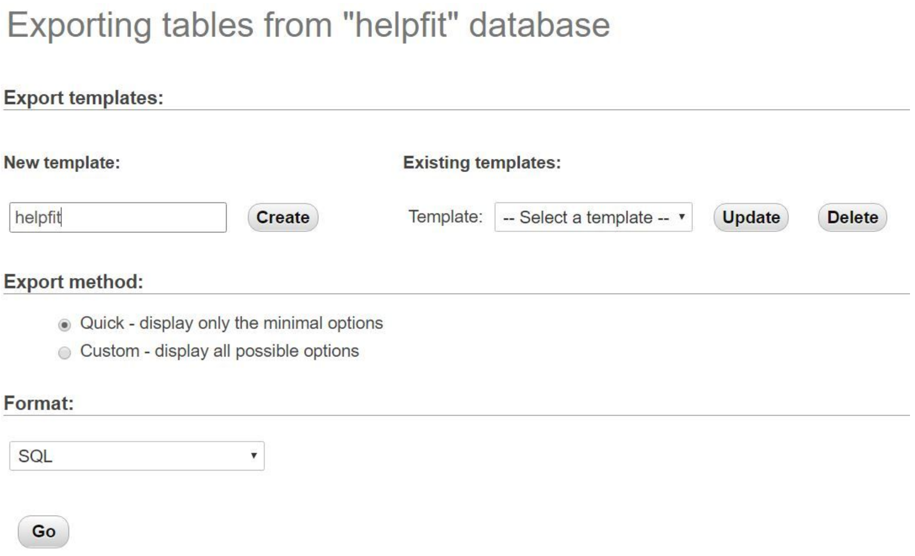
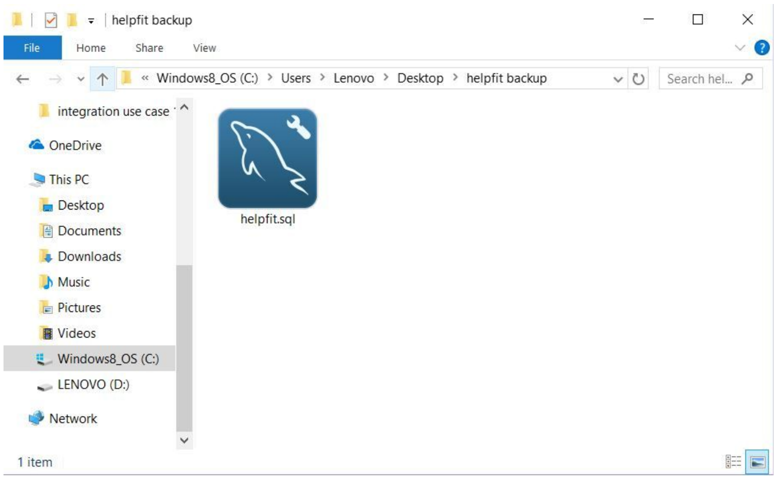
Updated Training History (Group)

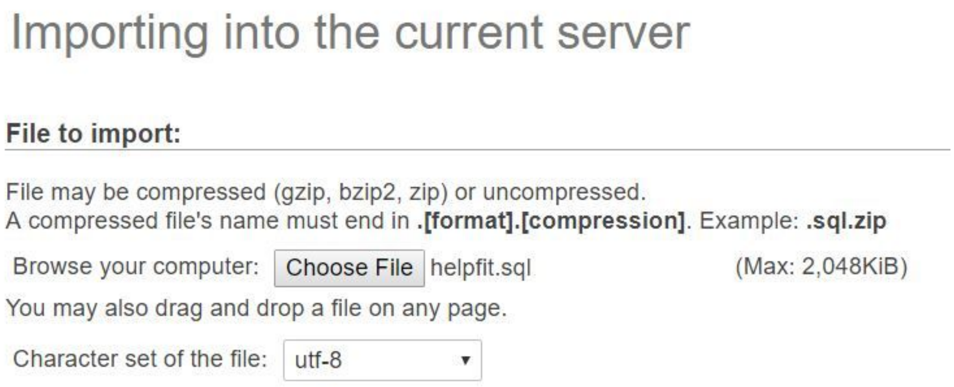
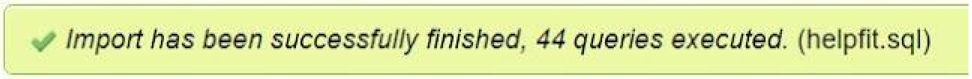


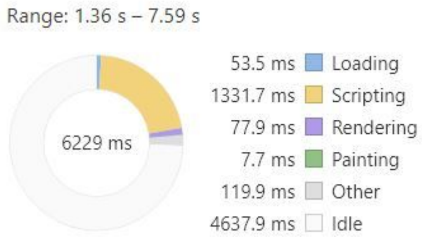
←T→	▼	sessionID	titel	date	time	feee	status	notes	sessionType	classType	maxParticipants	countPart	trainerID
□	Edit Copy Delete	33	Personal 1 - renamed	22 November 2017	14:02	105	Available	null	Personal Training Session	null	null	0	2
□	Edit Copy Delete	34	Group 1	20 November 2017	11:11	200	Available	null	Group Training Session	Sport	20	0	2
□	Edit Copy Delete	35	Personal 2	30 November 2017	11:11	120	Available	null	Personal Training Session	null	null	0	2
□	Edit Copy Delete	36	Group 2 - renamed	28 November 2017	14:22	225	Available	null	Group Training Session	MMA	30	0	2

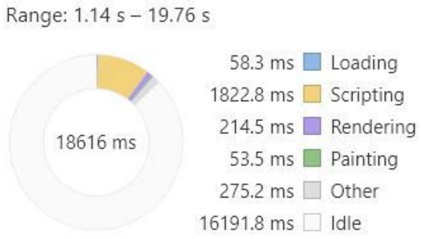
Updated ‘review’ table in database

9. System testing (Non-functional)

Test Case 1: Ability to backup data
Input
 Exporting tables from "helpfit" database Export templates: New template: helpfit Create Existing templates: Template: -- Select a template -- Update Delete Export method: ● Quick - display only the minimal options ○ Custom - display all possible options Format: SQL Go Exporting helpfit.sql from database to local
Expected outcome
A file named helpfit.sql will be exported and saved to local.
Actual outcome
 helpfit backup Windows8_OS (C:) › Users › Lenovo › Desktop › helpfit backup helpfit.sql 1 item File Home Share View Search hel... integration use case OneDrive This PC Desktop Documents Downloads Music Pictures Videos Windows8_OS (C:) LENOVO (D:) Network helpfit.sql successfully exported to local

Test Case 2: Ability to restore data
Input
 Importing into the current server **File to import:** File may be compressed (gzip, bzip2, zip) or uncompressed. A compressed file's name must end in **.[format].[compression]**. Example: **.sql.zip** Browse your computer: Choose File helpfit.sql (Max: 2,048KiB) You may also drag and drop a file on any page. Character set of the file: utf-8 ▾ Importing helpfit.sql from local to database
Expected outcome
The file named helpfit.sql will be imported to the database successfully. All the data will be restored.
Actual outcome
 ✓ *Import has been successfully finished, 44 queries executed.* (helpfit.sql) helpfit.sql successfully imported to database

Test case 3: Performance test (Use Case 5: Review Trainer)
Input
1. From Member Training History page click "Review" button. 2. Submit review form.
Expected outcome
Total loading time should be less than 2 seconds.
Actual outcome
 Range: 1.36 s – 7.59 s 6229 ms 53.5 ms 🟦 Loading 1331.7 ms 🟨 Scripting 77.9 ms 🟪 Rendering 7.7 ms 🟩 Painting 119.9 ms ⬜ Other 4637.9 ms □ Idle Performance profile results for Use Case 5: Review Trainer

Test case 4: Performance test (Use Case 6: Update Training Session)
Input
1. From Trainer History Page click "Edit" button. 2. Submit updated training session form.
Expected outcome
Total loading time should be less than 2 seconds.
Actual outcome
 Range: 1.14 s – 19.76 s 18616 ms 58.3 ms 🟦 Loading 1822.8 ms 🟨 Scripting 214.5 ms 🟪 Rendering 53.5 ms 🟩 Painting 275.2 ms ⬜ Other 16191.8 ms □ Idle Performance profile results for Use Case 6: Update Training Session

10. Compatibility test

Platform	**Platform version**	**Compatibility**
Web browsers		
Google Chrome	61.0.3163.100 (Official Build) (64-bit)	Yes
Internet Explorer	11.1770.14393.0	Yes
Mozilla Firefox	56.0.2 (64-bit)	Yes
Safari	Version 11.0 (12604.1.38.1.7)	Yes
Mobile phones		
iPhone 5	iOS 10.3.3	Yes
iPhone 6+	iOS 11.0.3	Yes
iPad	iOS 11.0.3	Yes
iPad Pro	iOS 11.1	Yes
Samsung Galaxy S5	Android 5.0	Yes
Nexus 6P	Android 7.1.2	Yes

11. Test analysis report

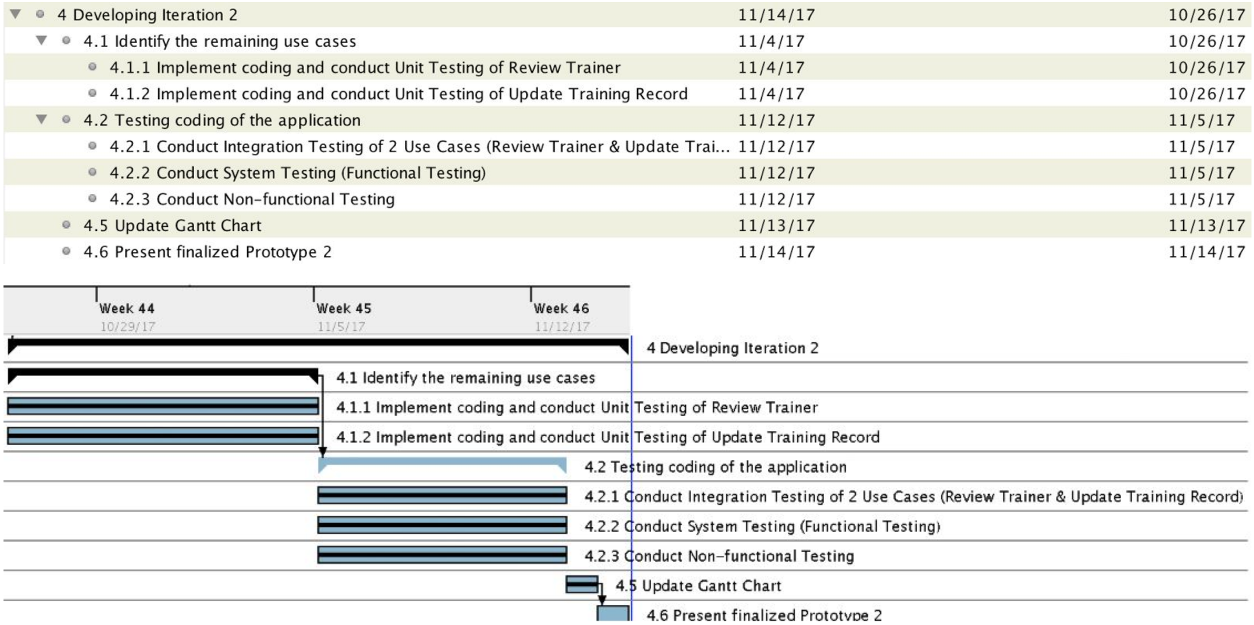
For Iteration 2, we have conducted a number of tests for each Use Case 5 (Review Trainer) and Use Case 6 (Update Training Session).

First, unit testing was conducted for each use case using the white box method to ensure each individual component of the system are tested. These unit tests are to validate that each unit of the system is performing as it was designed to.

Next, integration testing was conducted using the black box method. For these tests, individual units are combined and tested as a group. All the integration test cases carried out validates that there are no errors in the interaction of each module, concluding that our system is integrated properly.

After the unit and integration testings have completed, we proceeded to conduct system testing to test the completed and fully integrated system. All of the test cases have proven that each use case can be done without any faults and errors. This concludes that our system meets all of the requirements.

12. Updated Gantt chart



4 Developing Iteration 2	11/14/17	10/26/17
4.1 Identify the remaining use cases	11/4/17	10/26/17
4.1.1 Implement coding and conduct Unit Testing of Review Trainer	11/4/17	10/26/17
4.1.2 Implement coding and conduct Unit Testing of Update Training Record	11/4/17	10/26/17
4.2 Testing coding of the application	11/12/17	11/5/17
4.2.1 Conduct Integration Testing of 2 Use Cases (Review Trainer & Update Trai...	11/12/17	11/5/17
4.2.2 Conduct System Testing (Functional Testing)	11/12/17	11/5/17
4.2.3 Conduct Non-functional Testing	11/12/17	11/5/17
4.5 Update Gantt Chart	11/13/17	11/13/17
4.6 Present finalized Prototype 2	11/14/17	11/14/17

Week 44
10/29/17
Week 45
11/5/17
Week 46
11/12/17
4 Developing Iteration 2
4.1 Identify the remaining use cases
4.1.1 Implement coding and conduct Unit Testing of Review Trainer
4.1.2 Implement coding and conduct Unit Testing of Update Training Record
4.2 Testing coding of the application
4.2.1 Conduct Integration Testing of 2 Use Cases (Review Trainer & Update Training Record)
4.2.2 Conduct System Testing (Functional Testing)
4.2.3 Conduct Non-functional Testing
4.5 Update Gantt Chart
4.6 Present finalized Prototype 2

Updated Gantt Chart (Iteration 2)

13. Conclusion

Did the group meet the objectives defined in Assignment 1?

At the end of Iteration 2 of this assignment, we have successfully developed Project HELPFit Fitness Prototype 2 that meets the project objectives defined in Assignment 1. With Prototype 2, fitness enthusiasts do not have to make appointment with the fitness trainers to register for a training session. Making Project HELPFit Fitness to be a server based system eliminates the use of papers for registration process which also meets our one of our objectives. By carrying out the system testings, it is ensured that the application will be accessible and executable without failure of execution. We have also come up with a simple and clear design for the application so it is easy to learn and use, meeting all of our project's objectives.

What went wrong?

For the coding implementation section, everything looks fine when building the website and verifying them on the web browser to check if there are any errors exist. When it comes to implementing the validation part and having the unit testing in order to test run the code to check whether it is all executing well, it really takes a lot of time. One of the validation is when the date and time of the session had passed the current date and time, the session's status has to be set to "Completed". Once the status has been set to "Completed", the review button will be enabled on the member view history table and disabled if the session status is still "Available". At first, it is to use PHP scripts code to call the datetime function to set the current datetime and the date of the training session to be compared and have an Update SQL query to update the session status, but there is an issue which caused the database to update the session status of all rows of the training session to "Completed". It should only update the status of the records which the date has passed the current date. This is because the Update query condition was not set for the date comparison based on the data from database table but instead it was based on the condition set on the PHP script and if the condition match, it will update the status of the entire records. Hence, for the Update SQL query have to add on the condition where the date function of SQL and compare with the timestamp of the current time and type cast by the date function to have the condition matched as it will do the comparison of both and select the row of the data and update the status. This is by far the most difficult part of the code implementation on the assignment.

What went right?

In Iteration 1, we have done about 4 use cases and 2 remaining use cases on Iteration 2 and hence we have sufficient time to work on the documentation and testings for our website implementation compared to Iteration 1.

What you would have done differently?

If given the chance to do things differently for this assignment, we would have kickstarted the project earlier to finish the code implementation before the end of the planned coding phase so that we can keep on testing the website until it meets all the functional requirements. This is to ensure we have extra time for fixing the bugs and error. By doing this, we would be able to work on the extra the non-functional requirements to make the website more secure, reliable and efficient. Besides that, the planning phase takes the most important role while having this subject and everything should be well planned at the very beginning of the project. We would follow the schedule that is set during the planning phase and set the goals to be ahead of the schedule to avoid anything went wrong at last minute.