**Gym Reservation System**

**Test plan and implementation**

Prepared by:

Daler Karimov, Guillaume Reveillere, Hangil Kim,

Rashad Aliyev, Viktor Shin, Youngju Kim

CS408B Computer Science Project

KAIST

2010.05.08

Contents

[List Of Figures 3](#_Toc261028211)

[List of Tables 3](#_Toc261028212)

[1. Overview 4](#_Toc261028213)

[1.1 Introduction 4](#_Toc261028214)

[1.2 Goal 4](#_Toc261028215)

[1.3 Work Division 4](#_Toc261028216)

[2. Functionality Test 5](#_Toc261028217)

[2.1 Timetable test 5](#_Toc261028218)

[2.2 Delete busy slot test 7](#_Toc261028220)

[2.3 Authentication page test 9](#_Toc261028222)

[2.4 Login page test 11](#_Toc261028224)

[2.5 Making reservation page test 13](#_Toc261028226)

[2.6 Register page test 15](#_Toc261028228)

[2.7 Locking slot page test 18](#_Toc261028230)

[2.8 My reservation page test 21](#_Toc261028232)

[3. Use Case Test 25](#_Toc261028234)

[3.1 Login test 25](#_Toc261028235)

[3.2 Place reservation test 26](#_Toc261028236)

[3.3 Modify/remove reservation test 27](#_Toc261028237)

[3.4 Register test 28](#_Toc261028238)

[4. User Acceptance Test 29](#_Toc261028239)

[4.1. Introduction 29](#_Toc261028240)

[4.2 Result from User Acceptance Test 33](#_Toc261028241)

[4.3 Conclusion 34](#_Toc261028242)

# List Of Figures

[Figure 1. Control Flow Graph for Timetable 5](#_Toc260960576)

[Figure 2. Control Flow Graph for deletebusyslots 9](#_Toc260960577)

[Figure 3. Control Flow Graph for authentication 11](#_Toc260960578)

[Figure 4. Control Flow Graph for login 13](#_Toc260960579)

[Figure 5. Control Flow Graph for making reservation 15](#_Toc260960580)

[Figure 6. Control Flow Graph for register 17](#_Toc260960581)

[Figure 7. Control Flow Graph for Locking slot test 20](file:///C:\Users\Viktor\Desktop\Gym%20Reservation%20System_test%20plan_0506.docx#_Toc260960582)

[Figure 8. Control Flow Graph for MyReservation Test 23](file:///C:\Users\Viktor\Desktop\Gym%20Reservation%20System_test%20plan_0506.docx#_Toc260960583)

# List of Tables

[Table 1. Timetable test 1 6](#_Toc261988132)

[Table 2. Delete Busy slot test 1. 8](#_Toc261988133)

[Table 3. Authentication page test 10](#_Toc261988134)

[Table 4. Login page test 12](#_Toc261988135)

[Table 5. Making Reservation test 14](#_Toc261988136)

[Table 6. Register Page test 17](#_Toc261988137)

[Table 7. Locking Test 20](file:///C:\Users\Viktor\Desktop\Gym%20Reservation%20System_test%20plan_team%202.docx#_Toc261988138)

[Table 8. My reservation Test 1 22](#_Toc261988139)

[Table 9. My reservation Test 2. 23](#_Toc261988140)

[Table 10. My reservation Test 4. 24](#_Toc261988141)

[Table 11. Login Use Case Test 25](#_Toc261988142)

[Table 12. Place reservation Use Case Test 26](#_Toc261988143)

[Table 13. Modify/remove reservation Use Case Test 27](#_Toc261988144)

[Table 14. Register Use Case Test 28](#_Toc261988145)

# 1. Overview

## 1.1 Introduction

This document describes implementation and test plans that our team performed. We used PHP, JavaScript, CSS for coding and testing. After implementation we performed Unit, Use Case and User Acceptance test. We used structural coverage criteria for graph coverage during testing, our test requirements are determined by edge coverage. Also, for logic coverage, we considered clause coverage.

## 1.2 Goal

* To gain experience in test process
* To learn how to use different types of tests
* To find and fix bugs and errors in our implementation code

## 1.3 Work Division

|  |  |  |
| --- | --- | --- |
| **Responsibility** | **Implementation** | **Test** |
| Login/logout | Yeongju Kim | Rashad Aliyev |
| Register | Yeongju Kim | Yeongju Kim |
| Locking slot | Yeongju Kim | Yeongju Kim |
| Timetable | Rashad Aliyev | Rashad Aliyev |
| Pop-up with reservation information | Guillaume Reveillere | Yeongju Kim |
| MyReservation | Viktor Shin | Viktor Shin |
| Modify/Cancel reservation | Viktor Shin | Viktor Shin |
| Webpage design | Daler Karimov, Hangil Kim | Daler Karimov, Hangil Kim |
| Integration | Hangil Kim, Yeongju Kim |  |
| Authentication | Yeongju Kim | Yeongju Kim |
| User Acceptance Test | Yeongju Kim | Yeongju Kim |

# 2. Functionality Test

We did the functionality test instead of unit test, given that a php page is a unit and it has its own functionality.

## 2.1 Timetable test

### Figure 1. Control Flow Graph for Timetable

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **File(s)** | | | **Author** | | **Date** | **Version** | |
| timetable1.php | | | Rashad Aliyev | | 10-05-05 | 1.0.1 | |
| **Functionality Description** | | | | | | | |
| Prints the timetable for the given day in the given Gym. Information about existing reservations is displayed, and user is able to make reservation by clicking the button. The popup window is opened when clicked on the button. | | | | | | | |
| **Input/output Specifications** | | | | | | | |
| The page gets input parameters: $date and $facilityId from the page from where timetable1.php is called. It gets $userid from the session variables.  Output: FacilityID, Date, and hour of reservations are passed to next page. | | | | | | | |
| **Test Requirements** | | | | | | | |
| (0,1), (1,2), (2,3), (2,4), (4,5), (5,6), (6,8), (5,7), (7,8), (8,2) | | | | | | | |
| **Test Paths** | | | | | | | |
| 1. [0, 1, 2, 3]  2. [0, 1, 2, 4, 5, 6, 8]  3. [0, 1, 2, 4, 5, 7, 8] | | | | | | | |
| **No** | **Expected Result** | **Actual Result** | | **Comment** | | | **Pass** |
| **1** | Table is shown with information from the DB. On click user gets alert message about his status | All expectations are met | |  | | | O |
| **2** | Table is shown with information from the DB. On click popup window opened to make a reservation. | All expectations are met | |  | | | O |
| **3** | Empty table is shown | Empty Table is shown | |  | | | O |

#### Table 1. Timetable test 1

## 2.2 Delete busy slot test

### Figure 2. Control Flow Graph for deletebusyslots

DB Search Where Status = “Busy”

Busy slot doesn’t exist

Busy slot exists

Delete slots

CreateTime+TimeLimit<now

CreateTime+TimeLimit>now

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **File(s)** | | | **Author** | | **Date** | **Version** | |
| Deletebusyslots.php | | | Yeongju Kim | | 10-05-06 | 1.0.1 | |
| **Functionality Description** | | | | | | | |
| Deletebusyslots.php is included at the top of timetable1.php. Whenever timetable1.php is accessed by a user, it deletes every slots which have been remained ‘Busy’ for 60 seconds. | | | | | | | |
| **Input/Output Specifications** | | | | | | | |
| None | | | | | | | |
| **Test Requirements** | | | | | | | |
| (0, 1), (0, 2), (1, 2), (1, 3), (3, 4), (2, 4) | | | | | | | |
| **Test Paths** | | | | | | | |
| 1. [0, 2, 4]  2. [0, 1, 2, 4]  3. [0, 1, 3, 4] | | | | | | | |
| **No** | **Expected Result** | **Actual Result** | | **Comment** | | | **Pass** |
| **1** | No change in DB | No change in DB | |  | | | O |
| **2** | No change in DB | No change in DB | |  | | | O |
| **3** | Some rows which ‘Status’ == ‘Busy’ and ‘CreateTime’ is less than ‘current time’ - 60 | Some rows which ‘Status’ == ‘Busy’ and ‘CreateTime’ is less than ‘current time’ - 60 | |  | | | O |

#### Table 2. Delete Busy slot test 1.

## 2.3 Authentication page test

### Figure 3. Control Flow Graph for authentication

Get variable from session and cookie

Session Information == UserID

Not equal

Error Message

Error Message

Redirect to index.php

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **File(s)** | | | **Author** | | **Date** | **Version** | |
| Need\_auth.php | | | Yeongju Kim | | 10-05-06 | 1.0.1 | |
| **Functionality Description** | | | | | | | |
| Need\_auth.php checks if the user has the session and cookie information which is created when a logged in. Otherwise, redirect to index.php. | | | | | | | |
| **Input/Output Specifications** | | | | | | | |
| None | | | | | | | |
| **Test Requirements** | | | | | | | |
| (0, 1), (0, 2), (2, 3) | | | | | | | |
| **Test Paths** | | | | | | | |
| 1. [0, 1]  2. [0, 2, 3] | | | | | | | |
| **No** | **Expected Result** | **Actual Result** | | **Comment** | | | **Pass** |
| **1** | The user access the page normally which includes need\_auth.php | The user access the page normally which includes need\_auth.php | |  | | | O |
| **2** | An error message appeared, and redirect to index.php | An error message appeared, and redirect to index.php | |  | | | O |

#### Table 3. Authentication page test

## 2.4 Login page test

Get input from user

Search DB for UserID

UserID verified

UserID not verified

Password not verified

Password verified

Error

Store user’s information in the session and cookie

Error

### Figure 4. Control Flow Graph for login

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **File(s)** | | | **Author** | | **Date** | **Version** | |
| Proc\_login.php | | | Yeongju Kim | | 10-05-06 | 1.0.1 | |
| **Functionality Description** | | | | | | | |
| After a user enters ID and password, it compares them to corresponding row in User table. If verified, store the user information in session and cookie. Otherwise, redirect to index.php | | | | | | | |
| **Input/Output Specifications** | | | | | | | |
| Input: Userid and password from index.php  Output: None | | | | | | | |
| **Test Requirements** | | | | | | | |
| (0, 1), (1, 2), (1, 3), (3, 4), (3, 5) | | | | | | | |
| **Test Paths** | | | | | | | |
| 1. [0, 1, 2]  2. [0, 1, 3, 4]  3. [0, 1, 3, 5] | | | | | | | |
| **No** | **Expected Result** | **Actual Result** | | **Comment** | | | **Pass** |
| **1** | An error message says ‘no such userid in DB’ and redirects to index.php | An error message says ‘no such userid in DB’ and redirects to index.php | |  | | | O |
| **2** | An error message says ‘password not match’ and redirects to index.php | An error message says ‘password not match’ and redirects to index.php | |  | | | O |
| **3** | No error message. The user is logged on. | No error message. The user is logged on. | |  | | | O |

#### Table 4. Login page test

## 2.5 Making reservation page test

***Timetable.php***

***Locking the Slot***

User enters reservation information

Check if there are missing variables

Not missing

Values missing

DB Search

DB UserID == session UserID

DB UserID != session UserID

Error

Error

Update DB

### Figure 5. Control Flow Graph for making reservation

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **File(s)** | | | **Author** | **Date** | **Version** | |
| Input.php, proc\_confirm.php | | | Yeongju Kim | 10-05-06 | 1.0.1 | |
| **Functionality Description** | | | | | | |
| It assumes that a user clicks a time slot and popup is opened successfully. input.php is for entering reservation information. After some validation checks, it updates DB and the reservation is placed. If the user confirms the reservation 60 seconds after the popup is opened, the user cannot make a reservation. | | | | | | |
| **Input/Output Specifications** | | | | | | |
| Input: Date, StartTime, FacilityID  Output: None | | | | | | |
| **Test Requirements** | | | | | | |
| (0, 1), (1, 2), (2, 3), (3, 4), (4, 2), (3, 5), (5, 6), (5, 7), (7, 0) | | | | | | |
| **Test Paths** | | | | | | |
| 1. [0, 1, 2, 3, 5, 6]  2. [0, 1, 2, 3, 4, 2]  3. [0, 1, 2, 3, 4, 5, 7, 0] | | | | | | |
| **No** | **Expected Result** | **Actual Result** | | **Comment** | | **Pass** |
| **1** | The user successfully makes a reservation. The corresponding row in DB is updated by the information user entered and Status=’Reserved’ | The user successfully makes a reservation. The corresponding row in DB is updated by the information user entered and Status=’Reserved’ | |  | | O |
| **2** | An error message says ‘missing value’ and redirect to input.php. | An error message says ‘missing value’ and redirect to input.php. | |  | | O |
| **3** | An error message says ‘time duration finished’ and close the popup. | An error message says ‘time duration finished’ and close the popup. | |  | | O |

#### Table 5. Making Reservation test

## 2.6 Register page test

Enter information

IDok==N

IDok==Y && No missing Value

Information validation

Search DB for ID

ID exists

ID doesn’t exist

IDok=Y

Invalid E-mail Format

Error

Update DB

Password!=passwd\_check

Error

Missing value

Error

Error

### Figure 6. Control Flow Graph for register

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **File(s)** | | | **Author** | **Date** | **Version** | |
| Register.php, proc\_register.php | | | Yeongju Kim | 10-05-06 | 1.0.1 | |
| **Functionality Description** | | | | | | |
| It gets user information at register.php and send the information to proc\_register.php. proc\_register.php checks ID and password validation and gives proper error for unwanted input. Finally it inserts a row of user information in User table. | | | | | | |
| **Input/Output Specifications** | | | | | | |
| None | | | | | | |
| **Test Requirements** | | | | | | |
| (0, 1), (1, 2), (1, 3), (3, 6), (1, 9), (1, 4), (4, 7), (4, 8), (4, 5), (8, 1), (9, 1) | | | | | | |
| **Test Paths** | | | | | | |
| 1. [0, 1, 3, 2, 1, 4, 7]  2. [0, 1, 3, 6, 3, 2, 1, 4, 7]  3. [0, 1, 9, 1, 3, 2, 1, 4, 7]  4. [0, 1, 3, 2, 1, 4, 5, 1, 4, 7]  5. [0, 1, 3, 2, 1, 4, 8, 1, 4, 7] | | | | | | |
| **No** | **Expected Result** | **Actual Result** | | **Comment** | | **Pass** |
| **1** | The user successfully registers to the system. | The user successfully registers to the system. | |  | | O |
| **2** | During ID check, the user gets an error ‘ID exists’ and search for other userid. | During ID check, the user gets an error ‘ID exists’. | |  | | O |
| **3** | After the user clicks ‘submit’ button, the user gets an error ‘missing value’ and the system redirects to register.php. | After the user clicks ‘submit’ button, the user gets an error ‘missing value’ and the system redirects to register.php. | |  | | O |
| **4** | After the user clicks ‘submit’ button, the user gets an error ‘password not match’ and the system redirects to register.php. | After the user clicks ‘submit’ button, the user gets an error ‘password not match’ and the system redirects to register.php. | |  | | O |
| **5** | The user gets an error ‘unvaild email format’ and the system redirects to register.php. | The user gets an error ‘unvaild email format’ and the system redirects to register.php. | |  | | O |

#### Table 6. Register Page test

## 2.7 Locking slot page test

### Figure 7. Control Flow Graph for Locking slot test

***Timetable.php***

DB Search

Select duration and click slot

No row for the slot

Reserved slot

Busy slot

DB UserID check

DB UserID==Session UserID

DB search for previous slot

Duration == 1

Row doesn’t exist

DB UserID==Session userID

Previous Duration == 2

Selected Duration == 1

Selected Duration == 2

Insert Busy row into DB with duration=1

DB search for next slot

Busy Slot

DB UserID==Session userID

Row doesn’t exist

StartTime==23:00

StartTime!=23:00

Error

Error

Error

DB UserID check

Redirect to input.php

Insert Busy row into DB with duration=2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **File(s)** | | | **Author** | **Date** | **Version** | |
| Proc\_lock.php | | | Yeongju Kim | 10-05-06 | 1.0.1 | |
| **Functionality Description** | | | | | | |
| When a user clicks a time slot at the timetable, a popup of proc\_lock.php is opened and check if the user has the turn to make a reservation. This function is made to have the only first user who clicks a slot can input the reservation information.  If a user clicks a slot and there is no row in DB denoting the time slot, the system create a new row and update it as ‘Status’=’Busy’, ‘CreateTime’=’now()’. And the system keep other users accessing the busy slot. | | | | | | |
| **Input/Output Specifications** | | | | | | |
| Input: Date, StartTime, FacilityID  Output: None | | | | | | |
| **Test Requirements** | | | | | | |
| (0, 1), (1, 2), (1, 3), (1, 4), (3, 4), (3, 14), (4, 6), (6, 8), (6, 7), (7, 11), (8, 12),  (8, 9), (9, 10), (9, 13), (13, 3), (12, 0), (2, 0), (13, 10), (10, 11) | | | | | | |
| **Test Paths** | | | | | | |
| 1. [0, 1, 4, 6, 7, 11]  2. [0, 1, 3, 4, 6, 7, 11]  3. [0, 1, 4, 6, 8, 9, 10, 11]  4. [0, 1, 4, 6, 8, 9, 13, 10, 11]  5. [0, 1, 2, 0]  6. [0, 1, 4, 5, 0]  7. [0, 1, 3, 14, 0]  8. [0, 1, 4, 6, 8, 12, 0]  9. [0, 1, 4, 6, 8, 9, 2, 0]  10. [0, 1, 4, 6, 8, 9, 13, 14, 0] | | | | | | |
| **No** | **Expected Result** | **Actual Result** | | **Comment** | | **Pass** |
| **1** | The system creates ‘Busy’ row in DB with ‘duration’=1 and redirects to input.php | The system creates ‘Busy’ row in DB with ‘duration’=1 and redirects to input.php | |  | | O |
| **2** | The system redirects to input.php. No change in DB. | The system redirects to input.php. No change in DB. | |  | | O |
| **3** | The system creates ‘Busy’ row in DB with ‘duration’=2 and redirects to input.php | The system creates ‘Busy’ row in DB with ‘duration’=2 and redirects to input.php | |  | | O |
| **4** | The system creates ‘Busy’ row in DB with ‘duration’=2 and redirects to input.php | The system creates ‘Busy’ row in DB with ‘duration’=2 and redirects to input.php | |  | | O |
| **5** | An error message ‘reserved already’ appears. | An error message ‘reserved already’ appears. | |  | | O |
| **6** | An error message ‘other user accessing’ appears. | An error message ‘other user accessing’ appears. | |  | | O |
| **7** | An error message ‘other user accessing’ appears. | An error message ‘other user accessing’ appears. | |  | | O |
| **8** | An error message ‘cannot reserve 2 hours from 23:00’ appears. | An error message ‘cannot reserve 2 hours from 23:00’ appears. | |  | | O |
| **9** | An error message ‘reserved already’ appears. | An error message ‘reserved already’ appears. | |  | | O |
| **10** | An error message ‘other user accessing’ appears. Table 7. Locking Test | An error message ‘other user accessing’ appears. | |  | | O |

## 2.8 My reservation page test

### Figure 8. Control Flow Graph for MyReservation Test

Month (range 1 to 12)

Year

Check month

Month==12

Month!=12

Next Month=1

Next Year=Year+1

Next Month=Month+1

Next Year=Year

DB search for reservation

Print Result row

Modify

Delete

User chooses Delete of Modify

Pass parameters

Check Parameters

Update DB

Success

Fail

Success Notice

Fail Notice

All parameters exist

Missing

Update DB

Fail

Success

Fail Notice

Success Notice

Success

Fail

Next Row

No Next Row

Finish Row

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **File(s)** | | | **Author** | **Date** | **Version** | |
| confirm\_delete .php | | | Viktor Shin | 10-05-06 | 1.0.1 | |
| **Functionality Description** | | | | | | |
| Check if delete query was successful and specific slot was deleted from database | | | | | | |
| **Input/Output Specifications** | | | | | | |
| **Input:** Date and StartTime of reservation are passed as parameters from myreservation page  **Output:** Return result of the query as Boolean variable | | | | | | |
| **Test Requirements** | | | | | | |
| (0,1), (0,2), (1,3), (2,3), (3,4), (3,5), (4,6),(6,7), (6,8), (7,9), (7,10), (8,11), (8,12), (9,13), (9,14), (13,16), (14,16), (11,15), (12,15), (16,17), (15, 17), (17,4), (17,18), (5,18) | | | | | | |
| **Test Paths** | | | | | | |
| 1. (6,8,11,15,17) 2. (6,8,12,15,17) | | | | | | |
| **No** | **Expected Result** | **Actual Result** | | **Comment** | | **Pass** |
| **1** | DB update fails and result is shown to user | Notification about failure update is given | |  | | O |
| **2** | DB update succeeds and result is shown to user | Successfully update DB and notify user | |  | | O |

#### Table 8. My reservation Test 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **File(s)** | | | **Author** | **Date** | **Version** | |
| confirm\_modify .php | | | Viktor Shin | 10-05-06 | 1.0.1 | |
| **Functionality Description** | | | | | | |
| Check if modification query was successful and returns correct result | | | | | | |
| **Input/Output Specifications** | | | | | | |
| **Input:** Date, StartTime, Activity, E-mail, Phone, NumberOfPeople, GroupName are passed as parameters from myreservation page  **Output:** Return query result as Boolean | | | | | | |
| **Test Requirements** | | | | | | |
| (0,1), (0,2), (1,3), (2,3), (3,4), (3,5), (4,6),(6,7), (6,8), (7,9), (7,10), (8,11), (8,12), (9,13), (9,14), (13,16), (14,16), (11,15), (12,15), (16,17), (15, 17), (17,4), (17,18), (5,18) | | | | | | |
| **Test Paths** | | | | | | |
| 1. (6,7,9,13,16) 2. (6,7,9,14,16) 3. (6,7,10,16) | | | | | | |
| **No** | **Expected Result** | **Actual Result** | | **Comment** | | **Pass** |
| **1** | DB update fails and result is shown to user | Notification about failure update is given | |  | | O |
| **2** | DB update succeeds and result is shown to user | Successfully update DB and notify user | |  | | O |
| **3** | Notification about missing parameters is given | Expectation is met | |  | | O |

#### Table 9. My reservation Test 2.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **File(s)** | | | **Author** | **Date** | **Version** | |
| myreservation.php | | | Viktor Shin | 10-05-06 | 1.0.1 | |
| **Functionality Description** | | | | | | |
| 1. Get current date and derive next month and year 2. Display reservation for this month and next month | | | | | | |
| **Input/Output Specifications** | | | | | | |
| Input: User ID and password are taken from session variables  Output: table with reservation for this month and next month | | | | | | |
| **Test Requirements** | | | | | | |
| (0,1), (0,2), (1,3), (2,3), (3,4), (3,5), (4,6),(6,7), (6,8), (7,9), (7,10), (8,11), (8,12), (9,13), (9,14), (13,16), (14,16), (11,15), (12,15), (16,17), (15, 17), (17,4), (17,18), (5,18) | | | | | | |
| **Test Paths** | | | | | | |
| 1. (0,1,3,4,6) 2. (0,2,3,4,6) 3. (0 ,1, 3, 5, 18) 4. (0, 2, 3, 5, 18) | | | | | | |
| **No** | **Expected Result** | **Actual Result** | | **Comment** | | **Pass** |
| **1** | Values of this month and year and next month and year. Parameters for confirm or delete should be passed | Variables are shown properly, all parameters are passed | |  | | O |
| **2** | Values of this month and year and next month and year. Parameters for confirm or delete should be passed | Variables are shown properly, all parameters are passed | |  | | O |
| **3** | Table with reservations is shown | Table with user’s reservations is shown | |  | | O |
| **4** | Table with reservations is shown | Table with user’s reservations is shown | |  | | O |

#### Table 10. My reservation Test 4.

# 3. Use Case Test

## 3.1 Login test

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case** | | | User is able to login/logout ( SRS 4.2, pp.12) | | | | |
| **Author** | | | Viktor Shin | | **Date** | 2010-05-05 | |
| **Scenario** | | | | | | | |
| 1. User goes to main page. 2. User enters user ID and password 3. System Verifies ID and password 4. Systems create new session variables 5. User presses logout button 6. System goes to main page | | | | | | | |
| **No** | **Expected Result** | | | **Actual Result** | | **Comment** | **Pass** |
| **1.** | Shows main page | | | Shows main page | |  | O |
| **2.** | System asks for user ID and Password | | | Fields, where use can fill his ID and password | |  | O |
| **3.** | System verifies ID and Password | | | System verifies ID and Password | |  | O |
| **4.** | System creates session variables | | | System creates session variables | |  | O |
| **5.** | System provides “Logout” button | | | System provides “Logout” button | |  | O |
| **6.** | System deletes session variable | | | System destroys session variables | |  | O |
| **Overall** | | Passes every test | | | | | |

#### Table 11. Login Use Case Test

## 3.2 Place reservation test

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case** | | | Place reservation (SRS 4.3, pp. 13) | | | | |
| **Author** | | | Viktor Shin | | **Date** | 2010-05-05 | |
| **Scenario** | | | | | | | |
| 1. User selects a month and a date on which the user wants to make a reservation. 2. System displays a table of time slots for the date which differentiates available slots and reserved slots. 3. User selects start time for reservation and duration 4. In database status of the slot is updated as ‘locked’ in order to make sure the user who came first is entering information for the reservation. 5. User enters some information such as group name, number of members, and phone number. 6. User clicks Confirm button. 7. The reservation is placed and database of status of the slot is updated as ‘reserved’. 8. System redirects to Main page | | | | | | | |
| **No** | **Expected Result** | | | **Actual Result** | | **Comment** | **Pass** |
| **1.** | Shows timetable and asks to select gym | | | Shows timetable and asks to select gym | |  | O |
| **2.** | System displays timetable with available and reserved slots | | | System displays timetable with available and reserved slots | |  | O |
| **3.** | Waits for user to select StartTime and duration | | | On timetable user can select slot and duration | |  | O |
| **4.** | Updates status as “Busy” | | | Inserts empty slot with status “Busy” | |  | O |
| **5.** | System provides window to fill information | | | System opens pop-up for user to fill information | |  | O |
| **7.** | System updates database | | | Update query to database | |  | O |
| **8.** | System goes back to main page | | | System redirects to main page | |  |  |
| **Overall** | | Passes every test | | | | | |

#### Table 12. Place reservation Use Case Test

## 3.3 Modify/remove reservation test

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case** | | User changes or cancels reservation (SRS 4.4, pp.17) | | | |
| **Author** | | Viktor Shin | **Date** | 2010-05-05 | |
| **Scenario** | | | | | |
| 1. User goes to ‘My reservation’ page. 2. System displays the reservations of current month and next month, if any. 3. User chooses one of the reservations to modify or cancel.   3.1 Pop-up window with information  3.2 User inputs details   1. User confirms desire to do modification or cancellation. 2. System sends corresponding update information to the database of slot information. | | | | | |
| **No** | **Expected Result** | | **Actual Result** | **Comment** | **Pass** |
| **1.** | Shows MyReservation page | | Shows MyReservation page |  | O |
| **2.** | Table with user’s reservations | | Table with user’s reservations |  | O |
| **3.** | Shows “Modify” and “Remove” button | | Shows “Modify” and “Remove” button |  | O |
| **3.1** | Opens pop-up window with information about reservation | | Opens pop-up window with information about reservation | “Remove” pop-up window doesn’t have any information | O |
| **3.2** | Provides editable fields to user | | Provides editable fields to user | “Remove” pop-up window doesn’t have any information | O |
| **4.** | Asks confirmation | | Pop-up window for confirmation | “Modify” pop-up doesn’t ask for confirmation | O |
| **5.** | Updates database | | Updates database |  | O |
| **Overall** | Passes every test | | | | |

#### Table 13. Modify/remove reservation Use Case Test

## 3.4 Register test

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case** | | | New user is registered | | | | |
| **Author** | | | Viktor Shin | | **Date** | 2010-05-05 | |
| **Scenario** | | | | | | | |
| 1. User presses “register” button 2. System asks user’s information 3. User enters information 4. System asks user to validate ID 5. System checks if password and confirmed password are same 6. System saves user’s information in database 7. System redirects to main page | | | | | | | |
| **No** | **Expected Result** | | | **Actual Result** | | **Comment** | **Pass** |
| **1.** | System provides registration | | | “register” button is available | |  | O |
| **2.** | System displays fields for user to enter his information | | | System displays fields for user to enter information | |  | O |
| **3.** | Waits for user to enter information | | | Fields can be filled by user | |  | O |
| **4.** | To proceed, system asks user to validate ID | | | Pop-up window with ID validation | |  | O |
| **5.** | System checks if both passwords are same | | | System checks if both passwords are same | |  | O |
| **7.** | System updates database | | | Insert query is performed | |  | O |
| **Overall** | | Passes every test | | | | | |

#### Table 14. Register Use Case Test

# 4. User Acceptance Test

## 4.1. Introduction

In actual situation, Gym Reservation System must be accessed by many users simultaneously. User Acceptance Test is planned to check what happens given that situation. To control the large number of concurrent accesses, each DB access is locked by transaction. However, we are not able to see if the concurrency problem is well resolve through Functionality Test and Use Case Test. User Acceptance Test was the great test plan in that aspect. Moreover, because Gym Reservation System is aimed to be used by KAIST student, it needs to be user-friendly. User Acceptance Test can get the valuable feedbacks from actual users.

31 testers participated in the User Test at 12:00am 7th May. They tested the Gym Reservation System along with a given scenario and we let them write down the experience for each step. The scenario makes the users competitively try reservation of specific time slots:

i. Register to the system and log in.

ii. Wait until midnight when the reservation slots are open.

iii. As soon as the slots are open, try to reserve West Gym - Badminton Court - 22:00~24:00, 15th May.

iv. If failed, try other time slots of that day.

v. Try for other gyms and other dates.

vi. Go to MyReservation page.

vii. Modify your reservation information.

The document spread over the testers is attached at next page.

---------------------------------------------------------start of User\_Test.doc----------------------------------------------------

**KAIST 체육관 예약 시스템 사용자 테스트**

**KAIST Gym Reservation System User Test**

1. **소개**

**Introduction**

체육관 예약 시스템(이하 GRS)은 웹을 기반으로 KAIST의 체육 시설 예약을 할 수 있도록 하기 위하여 시작된 프로젝트입니다. Unit Test와 Integration Test를 거쳤으며, 다수의 사용자가 동시 접근시의 문제점을 발견하거나 개발자 이외의 학생들의 피드백을 받고자 사용자 테스트를 기획하게 되었습니다. 사용자께서는 아래의 시나리오에 따라 시스템을 경험해보고 버그 혹은 사용시 불편한 점을 적어주시면 됩니다.

Gym Reservation System (GRS) is a web-based reservation interface for KAIST gyms. It has undergone Unit Test and Integration Test, and it is now ready to get actual user’s feedback. It needs to be tested in the environment such that many users access simultaneously. Experience this system following the steps the scenario describes below, and please write down any bugs and inconvenient things of this system.

시나리오의 각 단계마다 여러분이 경험한 것들(웹사이트의 메세지)과 함께 특이한 점, 버그 혹은 불편한 점을 자세히 적어주시면 GRS를 테스트하는 데에 큰 도움이 될 것입니다.

It will be the great help if you give us the feedback for each step, such as messages from the website, weird things, bugs, and inconvenient things.

1. **시나리오**

**Scenario**

여러분은 5월 15일에 West Gym의 Badminton Court에서 10pm~12am 시간을 예약하고 싶어합니다. 그 시간을 얻지 못한 경우 그 날 언제라도 괜찮다고 생각합니다.

You (all the users) want to make a reservation for 15th May, West Gym – Badminton Court, 10pm ~ 12am. If you fail to get that slot, you think, anytime for the day is ok.

당신이 한 슬롯을 클릭해서 예약 정보를 입력하기 시작했을 때, 다른 사람은 그 슬롯에 접근하지 못하게 됩니다. 만약 60초 내에 예약을 완료하지 못한다면 그 기회는 다른 사람에게 넘어가게 됩니다.

When you click a slot and start entering the reservation information, other users cannot access the slot. If you don’t confirm the reservation in 60 seconds, someone else get the chance to access the slot.

1. <http://kaistgrs.inames.kr/test/> 로 가 주세요.

Click <http://kaistgrs.inames.kr/test/>.

첫인상

First impression

1. Register 버튼을 눌러 회원가입을 해주세요.

Click ‘Register’ button to register.

코멘트

Comment

1. 예약 가능한 시간(자정)이 될 때까지 기다려주세요.

Wait until the open time (midnight).

1. 5월 15일, West Gym – Badminton Court, 22~24시를 누구보다 먼저 예약해야 합니다.

Try to make a reservation for 15th May, West Gym – Badminton Court, 10pm ~ 12am. Remember only one person can get that time!

예약을 시도했을 때 나타난 메세지, 특이한 점, 성공여부 등

Messages you got, special things, and ‘succeed/fail’ when you tried

- Important -

1. 그 시간에 예약하지 못한 경우, 해당 날짜의 아무 시간이나 예약해 주세요.

If you failed, make a reservation for anytime for the day (15th May).

예약을 시도한 시간/장소, 특이한 점, 성공여부 등

When/Where you tried to reserve, special things, and ‘succeed/fail’

- Important -

1. 다른 날짜나 다른 체육관에서 예약을 해보세요.

Try other reservations for anytime and anywhere.

코멘트

Comment

1. My Reservation 버튼을 눌러 자신의 예약정보를 확인해보세요.

Click ‘My Reservation’ button to make sure your reservations.

코멘트

Comment

1. 예약 정보를 변경해보세요.

Modify your reservation information.

코멘트

Comment

감사합니다! Thank you!

---------------------------------------------------------end of User\_Test.doc----------------------------------------------------

## 4.2 Result from User Acceptance Test

|  |  |
| --- | --- |
| **Number of Testers** | 31 (25 KAIST students and 6 others)/(15 Koreans and 16 others) |
| **Test Date and Time** | 2010/05/07 12:00AM ~ |
| **Document Updated** | 2010/05/08 |
| **Author** | Yeongju Kim |

Below table describes the reported complains and feedbacks.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **User’s comment** | **Occurrence** | **Developer’s Comment** |
| **Errors** | Internet browser errors | 5 | It’s common in usual use of browsers, so we cannot consider it as the system’s error |
| Cannot Register. “Error in the information you typed. Please check again”. I cannot figure out what information gives the error. | 2 | We found this also. We are trying to find the reason. |
| “You cannot make a reservation. Time duration for you is finished” even if I confirm the reservation in 60 seconds | 1 | We found this also. We are trying to find the reason. |
| Javascript error below the browser when I clicked Gym list | 1 | It would be because the user didn’t select the date yet. |
| **GUI** | Hard to find ‘Register’ and ‘login’ buttons | 8 | Fixed. |
| Design too simple | 6 | Agreed. |
| There are many blank parts at first page, and I thought page is not fully open. | 4 |  |
| no ‘go back’ or ‘go home’ in ‘My Reservation’ page. | 3 | Fixed. |
| Hard to figure out how to use it. What is the first step after I log in? | 3 |  |
| Please make it complete (register page) | 3 |  |
| Layout is not proper for the user’s selection flow (Gym – calendar – time slot) | 2 |  |
| Cannot see the Gym list fully at the selection form | 2 | Fixed. |
| Improve the GUI, some pages need to make it more colorful and interesting. | 2 |  |
| In time table hours should be in format 8 am or 8:00, not just 8, 9. At first I assumed it is order, not time slots. | 1 |  |
| **Suggestions for convenience** | It would be good if I can change date and facility at ‘My Reservation’ page. | 3 |  |
| Need a user manual. | 2 |  |
| Annoying that I should enter the same information which I entered when register. | 2 |  |
| Confused with the word ‘duration’ | 1 |  |
| Should I enter ‘-‘s in phone number or not? (when register) | 1 |  |
| Authentication for register by student ID | 1 | We are trying to use Portal Login System. |
| Clock for counting down 60 seconds | 1 |  |
| Complain about ID check(if id ok, then shouldn’t press use it). (Register) | 1 |  |
| **Positive**  **(Cheer Up’s)** | This is what we really wanted! | 7 | Thank you. |

## 4.3 Conclusion

Firstly, there are few reports about errors/bugs. In particular, there is no concurrency problem even if more than 30 people access the same time slot simultaneously. We think the DB transaction lock works properly. For other minor errors, we are almost at the solution and they will be resolved.

There were many valuable feedbacks as for GUI. We will consider it seriously, because the Gym Reservation System is born to be used by KAIST students who are the testers.