**WIL Report Management System**

Test plan

­­­

By

**Ms. Phinthip Samutloiwon 552115050**

**Mr. Veerapat In-ongkarn 562115055**

Department of Software Engineering

College of Arts, Media and Technology

Chiang Mai University

Project Advisor

**Dr. Prompong Sugunnasil**

**Document History**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Document Name** | **Version** | **Status** | | **Date** | **Viewable** | **Reviewer** | **Responsible** |
| **Documents** | | | | | | | |
| WRMS- TestPlan\_V.0.1.docx | Created document  - Add Chapter I  - Add Chapter II - Add Chapter III  - Add Chapter IV | | Draft | 27-02-2017 | PS,VI | PS,VI | PS,VI |
| WRMS- TestPlan\_V.0.2.docx | - Edit Chapter II  - Edit Chapter III  - Edit Chapter IV | | Draft | 14-02-2017 | PS,VI | PS,VI | PS,VI |
| WRMS- TestPlan\_V.0.3.docx | - Edit Chapter III  - Edit Chapter IV | | Draft | 15-02-2017 | PS,VI | PS,VI | PS,VI |
| WRMS- TestPlan\_V.0.4.docx | - Edit Chapter I  - Edit Chapter II  - Edit Chapter III  - Edit Chapter IV | | Draft | 20-04-2017 | PS,VI | PS,VI | PS,VI |

**\*PS = Phinthip Samutloiwon  
\*VI = Veerapat In-ongkarn  
\*PSU= Prompong Sugunnasil**

Table of Contents

[Chapter I | Introduction 4](#_Toc477363352)

[1.1 Purpose 4](#_Toc477363353)

[1.2 Scope 4](#_Toc477363354)

[1.3 Acronyms and Definitions 4](#_Toc477363355)

[Chapter II | Test Procedure 5](#_Toc477363356)

[2.1 Test Objectives 5](#_Toc477363357)

[2.2 Test Duration 5](#_Toc477363358)

[2.3 Test Responsibility 5](#_Toc477363359)

[2.4 Test Strategy 5](#_Toc477363360)

[2.5 Result of Testing 5](#_Toc477363361)

[2.6 Test Environment 6](#_Toc477363362)

[2.6.1 Hardware 6](#_Toc477363363)

[2.6.2 Software 6](#_Toc477363364)

[Chapter III | Unit Test 7](#_Toc477363365)

[3.1 UserService Class 7](#_Toc477363366)

[UTC-01: getUserById(int id): User 7](#_Toc477363367)

[UTC-02: getUserByCode(String code): User 7](#_Toc477363368)

[UTC-03: getMentorsByStudentId(int id):User[] 8](#_Toc477363369)

[UTC-04: getSupervisprdByStudentId(int id):User[] 8](#_Toc477363370)

[UTC-05: getStudentsBySupervisorId(int id): User[] 8](#_Toc477363371)

[UTC-06: getStudentsByMentorId(int id): User[] 8](#_Toc477363372)

[UTC-07: setUserOfStudent(int userId, int studentId): User[] 8](#_Toc477363373)

[UTC-08: create(Object data): User 8](#_Toc477363374)

[UTC-09: update(Object data, int id): User 8](#_Toc477363375)

[UTC-10: delete(int id): User 8](#_Toc477363376)

[3.2 ProjectService Class 9](#_Toc477363377)

[UTC-11: getProjectById(int id): Project 9](#_Toc477363378)

[3.3 TaskService Class 9](#_Toc477363379)

[UTC-12: 9](#_Toc477363380)

[3.4 CommentService Class 9](#_Toc477363381)

[UTC-13: 9](#_Toc477363382)

[3.5 NotificationService Class 9](#_Toc477363383)

[UTC-14: 9](#_Toc477363384)

[3.6 UserController Class 9](#_Toc477363385)

[UTC-15: show(int id): User 9](#_Toc477363386)

[UTC-16: setUserOfStudent(int uid, int sid): User 9](#_Toc477363387)

[UTC-17: update(Object request): User 9](#_Toc477363388)

[UTC-18: getUserByCode(String code): User 9](#_Toc477363389)

[3.7 AuthController Class 9](#_Toc477363390)

[UTC-19: authenticate(Object request): User 9](#_Toc477363391)

[UTC-20: getAuthenticatedUser(): User 9](#_Toc477363392)

[UTC-21: register(Object request): User 9](#_Toc477363393)

[UTC-22: reset(Object request, int id): User 9](#_Toc477363394)

[3.8 ProjectController Class 9](#_Toc477363395)

[UTC-23: index(Object request): Project[] 9](#_Toc477363396)

[UTC-24: store(Object request): Project 9](#_Toc477363397)

[UTC-25: show(int id): Project 9](#_Toc477363398)

[UTC-26: update(Object request, int id): Project 9](#_Toc477363399)

[UTC-27: destroy(int id): Project 9](#_Toc477363400)

[UTC-28: getLogs(int id): TaskLog[] 9](#_Toc477363401)

[UTC-29: myProject(int id): Project[] 9](#_Toc477363402)

[3.9 TaskConteoller Class 9](#_Toc477363403)

[UTC-30: 9](#_Toc477363404)

[3.10 CommentController Class 10](#_Toc477363405)

[UTC-31: 10](#_Toc477363406)

[3.11 NotificationController Class 10](#_Toc477363407)

[UTC-32: 10](#_Toc477363408)

[Chapter IV | System Test 11](#_Toc477363409)

[4.1 Feature #1 11](#_Toc477363410)

[STC-01: Registration 11](#_Toc477363411)

[STC-02: Edit profile 12](#_Toc477363412)

[STC-03: Login 12](#_Toc477363413)

[STC-04: Logout 12](#_Toc477363414)

[STC-05: Add registration code 12](#_Toc477363415)

[4.2 Feature #2 12](#_Toc477363416)

[STC-06: View tasks 12](#_Toc477363417)

[STC-07: View Statistics 12](#_Toc477363418)

[STC-08: Add project 12](#_Toc477363419)

[STC-09: Delete project 12](#_Toc477363420)

[STC-10: Add task 12](#_Toc477363421)

[STC-11: Edit task 12](#_Toc477363422)

[STC-12: Delete task 12](#_Toc477363423)

[STC-13: Move task 12](#_Toc477363424)

[4.1 Feature #3 12](#_Toc477363425)

[STC-14: View comments 12](#_Toc477363426)

[STC-15: Add comment 12](#_Toc477363427)

[STC-16: Edit comment 12](#_Toc477363428)

[STC-17: Delete comment 12](#_Toc477363429)

[4.2 Feature #4 12](#_Toc477363430)

[STC-18: View weekly report 12](#_Toc477363431)

[STC-19: Generate report 12](#_Toc477363432)

[4.3 Feature #5 12](#_Toc477363433)

[STC-20: Receive web notification 12](#_Toc477363434)

[STC-21: Receive email notification 12](#_Toc477363435)

# **Chapter I | Introduction**

## **Purpose**

The purpose of test plan document is to ensure that all functional requirements are implemented as in SRS documentation as well as provide an approach of unit and system testing. This test plan also helps to manage changes in the early phase of the project since we got more information and revise the plan.

## **Scope**

The scope of this test document includes test objectives, test duration, Responsibility person, Test strategy, planning of unit and system testing with test data, and testing activities. WRMS implement both of white box testing and black box testing techniques. We implemented white box testing to control structure of testing design procedure to derive test cases. Black box testing uses for derive set of input data and possible conditions.

## **Acronyms and Definitions**

|  |  |
| --- | --- |
| **Acronyms** | **Definitions** |
| UTC | Unit Test Case |
| STC | System Test Case |
| SRS | Software Requirement Specification |
| WRMS | WIL Report Management System |

# **Chapter II | Test Procedure**

## **Test Objectives**

The objectives of testing WIL Report Management System are

1. Identify the faults – Finding the fault which cause the failure.
2. Correct and remove the faults – Fixing errors and remove faults.
3. Ensure that all functionalities are implemented correctly.

## **Test Duration**

|  |  |
| --- | --- |
| **Progress** | **Date and Duration** |
| Progress I | Perform date:  Duration: days |
| Progress II | Perform date:  Duration: days |
| Final Progress | Perform date:  Duration: days |

## **Test Responsibility**

|  |  |
| --- | --- |
| **Item** | **Responsibility** |
| Unit test | VI |
| Record of unit test | VI |
| System test | VI, PS |
| Record of system test | VI, PS |

## **Test Strategy**

WRMS performed testing strategy by the following steps:

1. Design test cases
2. Prepare test data
3. Determine the expected result
4. Perform testing
5. Record all results

## **Result of Testing**

The results are separate into two parts:

1. Actual result: The actual results after testing
2. Pass/ Fail result:
   1. Pass: The actual result which match with the expected result.
   2. Fail: The actual result which does not match with the expected result.

## **Test Environment**

### **2.6.1 Hardware**

1. **DELL Vostro**

Processor Intel Core i5-5200U (2.20 GHz)

Memory 4 GB DDR3L 500 GB 5400 RPM

1. **MacBook Pro (Retina, Mid 2014)**

Processor 2.5 GHz Intel Core i7

Memory 16 GB 1600 MHz DDR 3

### **2.6.2 Software**

1. Operating system Windows 8.1
2. Operating system macOS Sierra
3. phpunit

# **Chapter III | Unit Test**

## UserService Class

### getUserById(int id): User

**Test description:** Test getUserById(int id) method by input id of user as a parameter.

**Test data:** refer to **Appendix A** in “User” content.

**Result:** The result will return in console.

**Test case:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Case** | **Description** | **Input** | **Expected** |
| 1 | Test by input user id as a request parameter | id = 1 | Get a student name `Veerapat In-ongkarn ` and id=1 |
| 2 | Test by input user id as a request parameter | id = 2 | Get a mentor name ` Albert ` and id=2 |
| 3 | Text by input wrong-id parameter | Id = 10 | Get an empty object |
| 4 | Test by input empty parameter | empty object | Response status 404 |

### getNotifiesByUserId(int id): Notify[]

### getMyStatistic(int userId): Object

### getStatisticByUserID(int id): Object

### getUserByCode(String code): User

**Test description:** Test getUserByCode(int code) method by input code of user as a parameter

**Test data:** refer to **Appendix A** in “User” content.

**Result:** The result will return in console.

**Test case:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Case** | **Description** | **Input** | **Expected** |
| 1 | Test by input user code as a request parameter | code = 12345 | Get a mentor name ` Albert Einstein`, id=2 and code=12345 |
| 2 | Text by input wrong-code parameter | Id = 22222 | Get an empty object |
| 3 | Test by input empty parameter | empty object | Response status 404 |

### getMentorsByStudentId(int id):User[]

**Test description:** Test getMentorsByStudentId (int id) method by input id of student as a parameter.

**Test data:** refer to **Appendix A** in “User” content.

**Result:** The result will return in console.

**Test case:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Case** | **Description** | **Input** | **Expected** |
| 1 | Test by input student id as a request parameter | id = 1 | Get a mentors of student id = 1 |
| 2 | Test by input mentor id as a request parameter | Id = 2 | Get a empty array |
| 3 | Test by input empty parameter | empty object | Response status 404 |

### getSupervisprdByStudentId(int id):User[]

### setUserofStudent(int id, int sid): User

### deleteUserofStudent(int id, int sid): User

### create(object data): User

### createStudent(object data): User

### createSupervisor(object data): User

### createMentor(object data): User

### update(object data, int id): User

### getNotifiesByUserId(int id): Notify[]

## ProjectService Class

### show(int id): Project

### all(): Project[]

### getLogsByProject(int id): TaskLog[]

### store(Object request, int userId): Project

### update(Object request, int id): Project

### destroy(int id): Project

### getMyProjectsByUserId(int id): Project[]

### getProjectsByUserID(int id): Project[]

## TaskService Class

### 

## CommentService Class

### 

## NotificationService Class

### 

## UserController Class

### show(int id): User

### setUserOfStudent(int uid, int sid): User

### update(Object request): User

### getUserByCode(String code): User

## AuthController Class

### authenticate(Object request): User

### getAuthenticatedUser(): User

### register(Object request): User

### reset(Object request, int id): User

## ProjectController Class

### index(Object request): Project[]

### store(Object request): Project

### show(int id): Project

### update(Object request, int id): Project

### destroy(int id): Project

### getLogs(int id): TaskLog[]

### myProject(int id): Project[]

## TaskConteoller Class

### 

## CommentController Class

### 

## NotificationController Class

### 

# **Chapter IV | System Test**

## **Feature #1**

### **Registration**

**Description:** Test for URS-01. Visitor can register to use the services.

**Required data**: name, email, password, role

**Test script**:

1. Visitor clicks on “Register”.
2. Visitor inputs name, email, password, and role.
3. Visitor clicks “Submit” button.

**Test Cases**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Case** | **Test script no.** | **Description** | **Input** | **Expected result** |
| 1 | 2 | Register with non-exist account as a student | name = `Veerapat In-ongkarn`,  email = `Veerapat@xmail.com`,  password = `123456`,  role = `student` | - The new account will be created.  - The system displays a successful message. |
| 2 | 2 | Register with non-exist account as a mentor | name = `Albert Einstein`  email = `Einstein@xmail.com`,  password = `111111`,  role = `mentor` | - The new account will be created.  - The system displays a successful message.  - The code of mentor will be created. |
| 3 | 2 | Register with non-exist account as a supervisor | name = `Marie Curie`,  email = `Curie@xmail.com`,  password = `222222`,  role = `supervisor` | - The new account will be created.  - The system displays a successful message.  - The code of supervisor will be created. |
| 4 | 2 | Register with an account with existing email | name = `Test existing email`,  email = `Einstein@xmail.com`,  password = `111111`,  role = `mentor` | - The system displays an error message. |
| 5 | 2 | Register with email incorrect data | name = `Marie Curie`,  email = `Curie`,  password = `222222`”,  role = `supervisor` | - The system displays an error message. |

### Edit profile

### Login

### Logout

### Add registration code

## **Feature #2**

### View tasks

### View Statistics

### Add project

### Delete project

### Add task

### Edit task

### Delete task

### Move task

## **Feature #3**

### View comments

### Add comment

### Edit comment

### Delete comment

## **Feature #4**

### View weekly report

### Generate report

## **Feature #5**

### Receive web notification

### Receive email notification