|  |
| --- |
| C:\Users\Tsubaki Yukino\Desktop\FlyAwayPlus Project\Capture.JPG |
| **IVOLUNTEER** |
| **Test Plan** |
|  |
| |  |  |  | | --- | --- | --- | | **Ivolunteer** | | | | **Group Members** | Nguyen Minh Duc | SE03052 | | Nguyen Thac Son | SE03343 | | Nguyen Van Tam | SE02495 | | Vo Ton Phuc | SE03162 | | Pham Van Trong | SE03292 | | **Supervisor** | Mr. Bui Dinh Chien | | | **Project code** | IVSN | | |
|  |

**- Hanoi, 05/2016 –**

**RECORD OF CHANGE**

\*A-Added; M-Modified; D-Deleted

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Effective Date** | **Change Items** | **A\*M, D** | **Change Description** | **New Version** |
| 30/05/2016 | Test plan | A | Add new | 1.0 |
|  |  |  |  |  |

*Table 1: Record of change*

**SIGNATURE PAGE**

|  |  |  |
| --- | --- | --- |
| **ORIGINATOR** | Pham Van Trong |  |
|  | Team Member |  |
|  |  |  |
|  |  |  |
| **REVIEWERS** | Nguyen Minh Duc |  |
|  | Project Manager |  |
|  |  |  |
|  |  |  |
| **APPROVAL** | Bui Dinh Chien |  |
|  | Supervisor |  |
|  |  |  |

Table of Contents

[**1.** **INTRODUCTION** 5](#_Toc452498952)

[1.1. Purpose 5](#_Toc452498953)

[1.2. Definitions and Acronyms 5](#_Toc452498954)

[1.3. References 6](#_Toc452498955)

[1.4. Background information 6](#_Toc452498956)

[1.5. Scope of testing 6](#_Toc452498957)

[1.6. Constraints 8](#_Toc452498958)

[1.7. Risk list 8](#_Toc452498959)

[1.8. Training needs 9](#_Toc452498960)

[**2.** **REQUIREMENTS FOR TEST** 11](#_Toc452498961)

[2.1. Test item 11](#_Toc452498962)

[2.2. Acceptance Test Criteria 12](#_Toc452498963)

[**3.** **TEST STRATEGY** 14](#_Toc452498964)

[3.1. Test type 14](#_Toc452498965)

[3.1.1. Function Testing 14](#_Toc452498966)

[3.1.2. User Interface Testing 14](#_Toc452498967)

[3.1.3. Data and Database Integrity Testing 15](#_Toc452498968)

[3.2. Test stages 16](#_Toc452498969)

[**4.** **RESOURCES** 16](#_Toc452498970)

[4.1. Human resource 16](#_Toc452498971)

[**5.** **TEST ENVIRONMENT** 17](#_Toc452498972)

[5.1. Hardware 17](#_Toc452498973)

[5.2. Software 17](#_Toc452498974)

[5.3. Infrastructure 17](#_Toc452498975)

[**6.** **TEST MILESTONES** 18](#_Toc452498976)

[**7.** **DELIVERABLES** 18](#_Toc452498977)

# **INTRODUCTION**

## Purpose

The purpose of the document describes scopes of test and activities which need to be taken during test process of project. It addresses the following items: Scopes of Testing, Requirements for Testing, Test Strategy, Test Resources, Test Environment, Test Milestones and Deliverables

## Definitions and Acronyms

This section describes the definitions, terms, and acronyms that are used in software requirements specification.

|  |  |  |
| --- | --- | --- |
| **Acronym** | **Definition** | **Note** |
| IVSN | Ivolunteer Social Network |  |
| TL | Test Leader |  |
| PM | Project Manager |  |
| QA | Quality Assurance |  |
| SRS | Software Requirement Specification |  |
| TC | Test Case |  |
| TP | Test Plan |  |
| ST | System Test |  |
| IT | Integration Test |  |
| UT | Unit Test |  |
| GUI | Graphic User Interface |  |
| TR | Test Report |  |
| KLOC | 1000 line of code |  |

*Table 1-3: Definition and Acronyms*

## References

|  |  |  |  |
| --- | --- | --- | --- |
| Title/File name | Author | Version | Effective Date |
| FAP\_Software Requirement Specification\_v1.0\_EN | TamNV | v1.0 | 28/05/2016 |
| FAP\_Project Plan\_v1.0\_EN | DucNM | v1.0 | 9/05/2016 |

## Background information

The target of testing is ensured all functions will be run correctly as SRS description. In addition, restrict maximum of defect during the user access in the application. To do this target, website will have to:

* Passed the stages of testing: Unit Testing, Component Testing, Integration Testing, System Testing, Acceptance Testing
* Passed the types of testing: Function Testing, User Interface Testing , Data and Data Integrity Testing
* Run normally in required devices/browsers.

## Scope of testing

IVSN will be tested by 5 phases:

**Phase 1: Unit testing**

* Unit testing will be done by developers
* Developers user White Box Testing technique to do
* When executing unit testing, if any bugs are found, developers have to log bug on “Defect Log Management” file and fix it until it is correct.

*Rule for filling test result:*

|  |  |
| --- | --- |
| Test result pass | Pass |
| Test result fail | Fail |
| Do not test | Untested |
| Cannot test | N/A (Not available) |

**Phase 2: Component testing**

* After finishing unit testing, component testing will be performed by testers.
* Material are component test cases, low- level design
* Testers user Black Box Testing technique to do
* When executing component testing, if any bugs are found, testers have to log on “Defect Log Management” file and assign to developer fix it and redo this process until it is correct.

*Rule for filling test result:*

|  |  |
| --- | --- |
| Test result pass | Pass |
| Test result fail | Fail |
| Do not test | Untested |
| Cannot test | N/A (Not available) |

**Phase 3: Integration testing**

* After finishing component testing, integration testing will be performed by testers.
* Material are integration test cases, high- level design and test tools.
* Do test by flow of functions and items which have relation.
* When executing integration testing, if any bugs are found, testers have to log on “Defect Log Management” file and assign to developer fix it and redo this process until it is correct.

*Rule for filling test result:*

|  |  |
| --- | --- |
| Test result pass | Pass |
| Test result fail | Fail |
| Do not test | Untested |
| Cannot test | N/A (Not available) |

**Phase 4: System testing**

* After finishing integration testing and developers collect all functions and items, testers will be performed system testing, it means doing test whole system.
* Material area system test case, SRS
* If any bugs are found, developers have to fix and testers will verify them. System test is ended only when test cases are passed and no bug is found.

*Rule for filling test result:*

|  |  |
| --- | --- |
| Test result pass | Pass |
| Test result fail | Fail |
| Do not test | Untested |
| Cannot test | N/A (Not available) |

**Phase 5: Acceptance testing**

* Base on customer/user requirement specification, system is tested again, for ensure there is not lacking or mistake any requirement.
* If there is any problem, developers have to fix/update and tester will verify them.
* Acceptance testing is ended only when whole system met requirement specification.

## Constraints

* Time of testing is short
* Have more environments should be tested: Window XP, Window 8… and more browsers: Firefox 30 and Google Chrome 40… But Test team can not cover all

## Risk list

* Not enough time to write enough test cases, execute test or re-test for fixed bug.
* Tester can be ill during the testing phase

## Training needs

ISVN Project follows V-Model process:



**Figure 1: V-Model**

Testing progress is divided to 5 phases include: Unit test, Component test, Integration test, System test and Acceptance test

* Unit test:
  + Unit testing is used to verify a single minimal unit of source code. The purpose of unit testing is to isolate the smallest testable parts of IVSN and verify that they function properly in isolation.
  + Unit testing is the first level of testing and is perform prior to component testing
  + Unit testing will be done by developer.
* Component test:
  + Component testing is used to validate a combined many minimal units of source code.
  + Component testing is performed after unit testing and before integration testing
  + Component testing will be done by tester
* Integration test:
  + Integration testing is a level of the software testing process where individual units or component are combined and tested as a group.
  + The purpose is to expose faults in the interaction between integrated units.
  + Integration testing is performed after component testing
  + Integration testing will be done by tester
  + There are two methods of doing integration testing: Bottom-up Integration testing and Top Down Integration testing:

|  |  |
| --- | --- |
| No | Integration Testing Method |
| 1 | Bottom-up integration  This testing begins with unit testing, followed by tests of progressively higher-level combinations of units called modules. |
| 2 | Top- Down integration  This testing, the highest-level modules are tested first and progressively lower-level modules are tested after that |

**Table 1-3:** Integration test

* System test:
  + System Testing is a level of the software testing process where a complete, integrated system is tested
  + The purpose is to evaluate the system’s compliance with the specified requirements
  + System testing is performed after integration testing
  + System testing will be done by tester
* Acceptance test:
  + Acceptance testing is performed after system testing
  + Acceptance testing will be performed by the test leader and team leader.
  + The acceptance test will be done for a period of 1 weeks after completion of the system test process

# **REQUIREMENTS FOR TEST**

## Test item

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Group of function | Function | Actor | Glossary |
|  | Common | Search | Register user,  Guest |  |
|  | Search group |  |
|  | Search project |  |
|  | View public group |  |
|  | View public project |  |
|  | Donate |  |
|  | Register | Guest |  |
|  | Login | Register user |  |
|  | Logout |  |
|  | Profile | Edit profile | Register user |  |
|  | Change display name |  |
|  | Change cover |  |
|  | Change information |  |
|  | Post/comment | Create post on other timeline | Register user |  |
|  | Create post on own timeline |  |
|  | Comment on a post |  |
|  | Delete own comment |  |
|  | Follow a post |  |
|  | Post management |  |
|  | Share management |  |
|  | Project | View project | Register user |  |
|  | Create project |  |
|  | Join project |  |
|  | Out project |  |
|  | Follow a project |  |
|  | Unfollow a project |  |
|  | Report a project |  |
|  | Group | View Group | Register user |  |
|  | Create group |  |
|  | Join group |  |
|  | Out group |  |
|  | Follow a group |  |
|  | Unfollow a group |  |
|  | Report group |  |
|  | Other user | Send friend request | Register user |  |
|  | Accept friend request |  |
|  | Delete friend |  |
|  | Friend management |  |
|  | Chat |  |
|  | Report a user |  |
|  | Album | Create own album | Register user |  |
|  | Upload image to own album |
|  | Album management |
|  | Change album name |
|  | Delete own album |
|  | Delete own album’s image |
|  | Album | Change project album name | Project leader | Registered user created a project |
|  | Change project album cover |
|  | Change project album permission |
|  | Delete project album |
|  | Delete project image |
|  | Album | Change group album name | Group leader | Registered user created a group |
|  | Change group album cover |
|  | Change group album permission |
|  | Delete group album |
|  | Delete group image |
|  | Member | Create ablum in project/group | Group member, project member | Registered user joined a project , group |
|  | Create post in project/group |
|  | Invite friend to project/group |
|  | Upload image to ablum in project/group |
|  | View private in project/group |
|  | Sponsor | Approve donate request | Group leader |  |
|  | Deny Donate Request |
|  | Add Sponsor |

## Acceptance Test Criteria

* Criteria for Unit test of Development team, for Test team accepts to start testing:
  + Number of TC/KLOC: 40TC/KLOC
  + Number defects/KLOC: 3-4 defects/KLOC
  + Statement coverage: 100%
  + Branch coverage: 100%
  + Path coverage: 100%
* Criteria for Component test:
  + Number of TC/KLOC: 30 TC/KLOC
  + Number defects/KLOC: 2-3 defects/KLOC
* Criteria for Integration test:
  + Number of TC/KLOC: 30 TC/KLOC
  + Number defects/KLOC: 2-3 defects/KLOC
* Criteria for System test:
  + Number of TC/KLOC: 20 TC/KLOC
  + Number defects/KLOC: 4-6 defects/KLOC
* Criteria for Acceptance test:
  + Number defects/KLOC: 1-2 defects/KLOC
  1. Feature not to be tested
* The stable of website when do not connect internet.
* Over than 5000 users connect to system at the same time.

# **TEST STRATEGY**

## Test type

### Function Testing

* Functional testing is a type of software testing whereby the system is tested against the functional requirements/specifications.
* Functions are tested by feeding them input and examining the output. Functional testing ensure that the requirements are properly satisfied by the website. This type of testing is not concerned with how processing occurs, but rather, with the results of processing.
* During functional testing, [Black Box Testing](http://softwaretestingfundamentals.com/black-box-testing/) technique is used in which the internal logic of the system being tested is not known to the tester.

|  |  |
| --- | --- |
| **Test Objective:** | The type of this test is to ensure proper target-of-test functionality, including user interaction, all function defined in specification document implemented correctly. |
| **Technique:** | Executing each use case, use-case flow, or function, using valid and invalid data, to verify the following:  - The expected results occur when valid data is used.  - The appropriate error or warning messages are displayed when invalid data is used.  - Each business rule is properly applied. |
| **Completion Criteria:** | - All planned tests have been executed.  - All identified defects have been addressed and closed. |
| **Special Considerations:** | Testing may be stopped when  Time runs out  A certain number of defects found  Test coverage > 97%  Stop when testing becomes unproductive |

**Table 3-1:** Function Testing

### User Interface Testing

GUI testing is the process of ensuring proper functionality of the GUI for a given web and making sure it conforms to its written specifications.

GUI testing evaluates design elements such as layout, colors, [fonts](http://whatis.techtarget.com/definition/font), font sizes, labels, text boxes, text formatting, captions, buttons, lists, icons, links, content and more.

|  |  |
| --- | --- |
| **Test Objective:** | Verify the following:  - Navigation through the target-of-test properly reflects business functions and requirements, including window-to-window, field-to-field, and use of access methods (tab keys, mouse movements, accelerator keys)  - Window objects and characteristics, such as menus, size, position, state, and focus conform to standards. |
| **Technique:** | Create or modify tests for each window to verify proper navigation and object states for each application window and objects. |
| **Completion Criteria:** | Each window successfully verified to remain consistent with benchmark version or within acceptable standard |
| **Special Considerations:** | Not all properties for custom and third party objects can be accessed. |

**Table 3-2:** GUI Testing

### Data and Database Integrity Testing

The databases and the database processes should be tested as a subsystem within the Project. These subsystems should be tested without the target-of-test’s User Interface as the interface to the data.  Additional research into the Database Management System (DBMS) needs to be performed to identify the tools and techniques that may exist to support the testing identified below.

|  |  |
| --- | --- |
| **Test Objective:** | Ensure database access methods and processes function properly and without data corruption. |
| **Technique:** | - Invoke each database access method and process, seeding each with valid and invalid data or requests for data.  - Inspect the database to ensure the data has been populated as intended, all database events occurred properly, or review the returned data to ensure that the correct data was retrieved for the correct reasons. |
| **Completion Criteria:** | All database access methods and processes function as designed and without any data corruption. |
| **Special Considerations:** | - Testing may require a DBMS development environment or drivers to enter or modify data directly in the databases.  - Processes should be invoked manually.  - Small or minimally sized databases (limited number of records) should be used to increase the visibility of any non-acceptable events. |

**Table 3-3:** Data and Data Integrity Testing

## Test stages

Clearly state the stage in which the test will be executed. Identified below are the stages in which common test are executed

| **Type of Tests** | **Stage of Test** | | | |
| --- | --- | --- | --- | --- |
| **Unit** | **Component** | **Integration** | **System** |
| Function Testing | X | X | X | X |
| User Interface Testing |  | X | X |  |

# **RESOURCES**

## Human resource

|  |  |  |
| --- | --- | --- |
| **Worker/Doer** | **Role** | **Specific Responsibilities/Comments** |
| TrongPV | Test Leader | Manage Test Resource and assign test tasks.  Create and review Test Plan.  Create and review Test Case.  Execute test.  Create and review Test Report |
| TamNV | Tester | Create and review Test Case.  Execute test.  Create Test view points  Create and review Test Report |

**Table 4-1:** Human resource

# **TEST ENVIRONMENT**

## Hardware

|  |  |  |
| --- | --- | --- |
| **Name** | **Purpose** | **Detail** |
| Laptop HP | Device for create and execute test | Window 7 Ultimate Core i5 |
| Laptop Dell | Device for create and execute test | Window 10 |

**Table 5-1:** Hardware

## Software

|  |  |  |
| --- | --- | --- |
| Name | Purpose | Detail |
| Test Plan | Managing test | Microsoft Word 2013 |
| Test case | Executing test | Microsoft Excel 2013 |
| Test report, Test checklist | Tracking test | Microsoft Excel 2013 |
| Chrome | Executing test | Chrome 40 or above |
| Firefox | Executing test | Firefox 30 or above |

**Table 5-2:** Software

## Infrastructure

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Purpose** | **Detail** | **Vendor/In-house** | **Version** |
| IVSN\_Defect Log Management\_v1.0\_EN | Tracking bug during testing time | Microsoft Excel 2013, 2010 | FPT-University | v1.0 |
| Test Effort | Effort execute test | Microsoft Excel 2013, 2010 | FPT-University | v1.0 |

**Table 5-3:** Infrastructure

# **TEST MILESTONES**

|  |  |  |  |
| --- | --- | --- | --- |
| **Milestone Task** | **Effort (pd)** | **Start Date** | **End Date** |
| Create Test Plan | 6 | 30/05/2015 | 03/06/2015 |
| Review & Update Test Plan | 2 | 04/06/2015 | 04/06/2015 |
| Create Component Test case | 12 | 05/06/2015 | 13/06/2015 |
| Review & Update Component Test Case | 2 | 13/06/2015 | 14/06/2015 |
| Create Integration Test case | 2 | 03/07/2015 | 09/07/2015 |
| Review & Update Integration Test Case | 2 | 14/07/2015 | 16/07/2015 |
| Create System Test case | 6 | 17/07/2015 | 21/07/2015 |
| Review & Update System Test Case | 2 | 22/07/2015 | 23/07/2015 |
| Execute Component test phase 1 | 1 | 31/07/2015 | 03/08/2015 |
| Execute Component test phase 2 | 1 | 12/08/2015 | 13/08/2015 |
| Execute Integration test phase 1 | 2 | 04/08/2015 | 05/08/2015 |
| Execute Integration test phase 2 | 2 | 14/08/2015 | 17/082015 |
| Execute System test phase 1 | 4 | 06/08/2015 | 07/08/2015 |
| Execute System test phase 2 | 4 | 18/08/2015 | 19/08/2015 |

**Table 6.1:** Test milestones

# **DELIVERABLES**

| **No** | **Deliverables** | **Responsibilities** | **Delivered Date** |
| --- | --- | --- | --- |
| 1 | Test Plan | Tester | 04/06/2015 |
| 2 | Component Test case | Tester | 15/06/2015 |
| 3 | Integration Test case | Tester | 16/07/2015 |
| 4 | System Test case | Tester | 23/07/2015 |
| 5 | Defect Log Management | All members | 24/08/2015 |
| 6 | Test report | PM | 24/08/2015 |

**Table 7-1:** Deliverables