|  |
| --- |
|  |

BUS USER SUPPORT SYSTEM

TEST PLAN

Project Code: BUSS

Document Code: BUSS\_Test\_Plan\_v1.0\_EN

**Ha Noi, 26/06/2015**

Record of change

\*A - Added M - Modified D - Deleted

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Effective Date** | **Changed Items** | **A\* M, D** | **Change Description** | **New Version** |
| 26/06/2015 | Add this document | A | Create document | v1.0 |
|  |  |  |  |  |

SIGNATURE PAGE

**AUTHOR:** Trinh Thi Tuyet Mai 26/06/2015

Test Leader (TL)

**REVIEWERS:** Nguyen Thanh Nam 29/06/2015

Project Manager (PM)

**APPROVAL:** Nguyen Van Sang 30/06/2015

Supervisor

TABLE OF CONTENTS

[1 INTRODUCTION 5](#_Toc423646971)

[1.1 Purpose 5](#_Toc423646972)

[1.2 Definitions, Acronyms, and Abbreviations 5](#_Toc423646973)

[1.3 References 5](#_Toc423646974)

[1.4 Background information 6](#_Toc423646975)

[1.5 Scope of testing 6](#_Toc423646976)

[1.6 Constraints 8](#_Toc423646977)

[1.7 Risk list 8](#_Toc423646978)

[1.8 Training needs 8](#_Toc423646979)

[2 Requirements for Test 10](#_Toc423646980)

[2.1 Test items 10](#_Toc423646981)

[2.2 Acceptance Test Criteria 11](#_Toc423646982)

[2.3 Feature not to be tested 12](#_Toc423646983)

[3 TEST STRATEGY 13](#_Toc423646984)

[3.1 Test types 13](#_Toc423646985)

[3.1.1 Function Testing 13](#_Toc423646986)

[3.1.2 User Interface Testing 14](#_Toc423646987)

[3.1.3 Data and Database Integrity Testing 14](#_Toc423646988)

[3.2 Test stages 15](#_Toc423646989)

[4 RESOURCE 16](#_Toc423646990)

[4.1 Human Resource 16](#_Toc423646991)

[5 Test environment 17](#_Toc423646992)

[5.1 Hardware 17](#_Toc423646993)

[5.2 Software 17](#_Toc423646994)

[5.3 Infrastructure 17](#_Toc423646995)

[6 TEST MILESTONES 19](#_Toc423646996)

[7 DELIVERABLES 20](#_Toc423646997)

# 

# INTRODUCTION

## Purpose

The purpose of this document is showing the scope of testing, test items, test strategy, testing approach, resources, test environment, schedule of intended testing activities and others related to testing.

## Definitions, Acronyms, and Abbreviations

| **Abbreviations** | **Description** | **Note** |
| --- | --- | --- |
| BUSS | Bus User Support System |  |
| SRS | Software Requirement Specification |  |
| TP | Test Plan |  |
| TC | Test Case |  |
| CT | Component Test |  |
| ST | System Test |  |
| IT | Integration Test |  |
| UT | Unit Test |  |
| GUI | Graphical User Interface |  |
| KLOC | 1000 line of code |  |

**Table 1-1**: Definitions and acronyms

## References

| **Title/File name** | **Author** | **Version** | **Effective Date** |
| --- | --- | --- | --- |
| BUSS\_Software\_Requirement\_Specification\_v1.0\_EN | BUSS Team | v1.0 | 19/06/2015 |
| BUSS\_Project\_Plan\_v1.0\_EN | NamNT | v1.0 | 19/05/2015 |

**Table 1-2:** Reference files

## 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.

## Scope of testing

* BUSS will be tested by 5 phases:

**Phase 1: Unit testing**

* Unit testing will be done by developers using White Box Testing technique
* When executing unit test, if any bugs are found, developers have to log bug into “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**

* Component testing will be performed by testers using Black Box Testing technique.
* Material are component test cases based on low- level design.
* When executing component test, if any bugs are found, testers have to log bug into “Defect Log Management” file and assign developer to fix it and repeat 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**

* Integration testing will be performed by testers.
* Material are integration test cases, high- level design and test tools.
* Doing test by checking flows between functions and items which have relation.
* When executing integration test, if any bugs are found, testers have to log on “Defect Log Management” file and assign developer to fix it and repeat 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, developers collect all functions and items then testers will perform system test, which means test the whole system.
* Materials are system test case, SRS
* If any bugs are found, developers have to fix then testers will verify them. System test is complete only when all 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 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 complete only when whole system met requirement specification.

## Constraints

* There are only 1 testers and time for Testing is short
* There are many screen size of mobile should be tested, but Tester can’t cover all

## Risk list

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

## Training needs

BUSS 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 BUSS 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 people outside project team.
* The acceptance test will be done for a period of 1 weeks after completion of the system test process.

# Requirements for Test

## Test items

| **No** | **Component** | **Feature to be tested** |
| --- | --- | --- |
| 1 | User Interface | Check the display of elements |
| 2 | Function | * Sign up * Log in/ Log out * Forgot password * Change password * Find path * Get current location * Display history of finding paths * Filter path found * Display path detail * Navigate route * Display bus list * Search bus * Display bus details: info, route, nearest, map * Display buses near current location * Set tracker * Cancel tracker * Display favorite buses * Add favorite bus * Delete favorite bus * Search favorite bus * Search place * Display favorite places * Add favorite place * Edit favorite place * Delete favorite places * Search favorite places * Display bus reminders * Add reminder * Edit reminder * Delete reminder * Suspend/Unsuspend reminder * Suspend/Unsuspend all reminder * Display next reminder * Settings |

**Table 2-1:** Test item

## Acceptance Test Criteria

* Criteria for Unit test of Development team, for Test team accepts to start testing:
* Number of UTC/KLOC: 40 UTC/KLOC
* Number defects/KLOC: 3-4 defects/KLOC
* Statement coverage: 97%
* Branch coverage: 100%
* Path coverage: 100%
* Criteria for Integration test:
* Number of UTC/KLOC: 30 UTC/KLOC
* Number defects/KLOC: 2-3 defects/KLOC
* Criteria for System test:
* Number of UTC/KLOC: 60 UTC/KLOC
* Number defects/KLOC: 4-6 defects/KLOC
* Criteria for Acceptance test:
* Number defects/KLOC: 1-2 defects/KLOC

## Feature not to be tested

* The stability and availability of GPS.
* The stability and availability of system without connecting to the network (3G, Wi-Fi ...)
* The stability and availability of hardware
* The location is outside of Hanoi
* Many users connect to system at the same time.

# TEST STRATEGY

## Test types

### Function Testing

* Functionality testing is performed to verify that a software application performs and functions correctly according to the functional requirements/specifications.
* Functions are tested by feeding them input and examining the output. 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.
* Functional testing typically involves these steps:
  + The identification of functions that the software is expected to perform
  + The creation of input data based on the function's specifications
  + The determination of output based on the function's specifications
  + The execution of the test case
  + The comparison of actual and expected outputs

|  |  |
| --- | --- |
| Test Objective | 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 testing a product's graphical user interface to ensure it meets its 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 | User Acceptance |
| Function Testing | X | X | X | X | X |
| User Interface Testing | X |  |  | X |  |
| Data & Data Integrity Testing |  |  | X | X |  |

# 

**Table 3-4:** Test stage

# RESOURCE

## Human Resource

|  |  |  |
| --- | --- | --- |
| Worker/Doer | Role | Specific Responsibilities/Comments |
| Trinh Thi Tuyet Mai | Tester | * Create Test Plan. |
| Bui Bich Phuong | 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 (Asus, Vaio, Dell) | Device for creating and executing test | Window 8.1 Pro |
| Mobile (Sony Z3, Zenphone 5, Samsung Note 3) | Device for executing test | Android 5.0+ |
|  |  |  |

**Table 5-1:** Hardware

## Software

|  |  |  |
| --- | --- | --- |
| Name | Purpose | Detail |
| Test Plan | Managing test | Microsoft Word 2013, 2010 |
| Test case | Executing test | Microsoft Excel 2013, 2010 |
| Test report, Test checklist | Tracking test | Microsoft Excel 2013, 2010 |
| Android Studio with Junit | Executing test | Android Studio with Junit |

**Table 5-2:** Software

## Infrastructure

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Purpose | Detail | Vendor/In-house | Version |
| BUSS\_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 Word 2013, 2010 | FPT-University | v1.0 |

**Table 5-3:** Infrastructure

# TEST MILESTONES

|  |  |  |  |
| --- | --- | --- | --- |
| Milestone Task | Effort (pd) | Start Date | End Date |
| Create TP | 5 | 16/06/2015 | 29/06/2015 |
| Review & update TP | 2 | 29/06/2015 | 30/06/2015 |
| Create Component Test case | 4 | 30/06/2015 | 03/07/2015 |
| Review & update Component TC | 2 | 06/07/2015 | 07/07/2015 |
| Create Integration TC | 2 | 07/07/2015 | 09/07/2015 |
| Review & Update Integration TC | 2 | 09/07/2015 | 11/07/2015 |
| Create System TC | 2 | 11/07/2015 | 13/07/2015 |
| Review & Update System TC | 2 | 13/07/2015 | 15/07/2015 |
| Execute CT phase 1 | 10 | 08/07/2015 | 23/07/2015 |
| Execute CT phase 2 | 5 | 29/07/2015 | 06/08/2015 |
| Execute IT phase 1 | 1 | 23/07/2015 | 24/07/2015 |
| Execute IT phase 2 | 1 | 06/08/2015 | 07/08/2015 |
| Execute ST phase 1 | 1 | 27/07/2015 | 28/07/2015 |
| Execute ST phase 2 | 1 | 10/08/2015 | 10/08/2015 |

**Table 6-1:** Test milestones

# DELIVERABLES

| No | Deliverables | Language | Delivered Date |
| --- | --- | --- | --- |
|  | Test Plan | English | 29/06/2015 |
|  | Component Test case | English | 08/07/2015 |
|  | Integration Test case | English | 27/07/2015 |
|  | System Test case | English | 05/08/2015 |
|  | Defect Log Management | English | 17/08/2015 |
|  | Test report | English | 17/08/2015 |

**Table 7-1**: Deliverables