American International University-Bangladesh



**FINAL PROJECT**

**Developing a Test Plan for Automated Ticket Issuing System for Dhaka**

Presented By

|  |  |
| --- | --- |
| Name | ID |
| Ferdous, Jannatul | 17-34717-2 |
| Rahi, Rahi Bulbul | 17-35696-3 |
| Chowdhury, Saba Zarin | 17-34788-2 |
| Anjum, MD Tanvir | 17-33647-1 |
| Mutasim Mahmud | 17-35735-3 |

Course: Software Quality and Testing

Course Teacher: S.M. ABDUR BHUIYAN ROUF

Section: B

Semester: Fall 2020-2021

CONTENT

|  |  |  |
| --- | --- | --- |
| .Serial | Criteria | Page No. |
| 1 | Test plan identifier | 3 |
| 2 | Introduction | 3 |
| 3 | References | 3 |
| 4 | Test Items | 3 |
| 5 | Software risk issues | 4 |
| 6 | Features to be tested | 4 |
| 7 | Features not to be tested | 5 |
| 8 | Approach | 5 |
| 9 | Item pass / fail criteria | 10 |
| 10 | Suspension criteria and resumption requirements | 10 |
| 11 | Test deliverables | 11 |
| 12 | Remaining test tasks | 11 |
| 13 | Environment needs | 12 |
| 14 | Roles and Responsibilities | 13 |
| 15 | Scheduling and estimation | 15 |
| 16 | Risk and Mitigation Plan | 15 |
| 17 | Conclusion | 16 |
|  |  |  |
|  |  |  |

**Test plan identifier**

**AIUB IT Solutions Inc. release 0.0.0.1 Software Test Plan.**

Note, the structure of this document is primarily based on the IEEE 829-1998 Standard for Software. Test Documentation. Additional reference standards include IEEE 1008 (Unit Testing), 1012 & 1059(Validation & Verification) and 1074 (Software Life Cycle process).

**References:**

TEST PLAN OUTLINE (IEEE 829 FORMAT)

<http://ece.uprm.edu/~icom5047/documents/OtherDocuments/TestPlan.pdf>

**Introduction**

This document is an overview showing our testing strategy for Automated Ticket Issuing System for Dhaka subway system. The goal is to provide automated ticket selling for the general public of Dhaka. This document will address the different standards that will apply to the unit, integration and system testing of the specified application. We will utilize testing criteria under the white box, black box, and system-testing paradigm. Throughout the testing process we will maintain coding standards and keep good communication between units. Check performance, stability and functionality. Also code quality assurance and system quality assurance & acceptance.

**Test Items**

The major items and functionalities of the system are as follows:

* 24/7 system service.
* Screen touch functionality in the menu selection.
* Source and destination selection.
* Ticket availability information on the display.
* Limit the amount of ticket issued at one time.
* Multiple Ticket issue in one transaction.
* Train arrival & departure time on display.
* Credit card/Debit card transactions.
* Coin/Taka recognition and acceptance
* Cancellation of ticket at any time during any transaction.

**Software risk issues**

* Touch screen functionality in menu selection should be tested precisely.
* Ticket availability function.
* Transaction functionality and safety issues in the system.
* Checking the complex functions of the system.
* Misunderstanding of system requirements.
* Change request for the system.
* Matching older version of product with the newer versions.
* Matching government regulation and rules with the system.

**Features to be tested**

At Dhaka Subway Systems, we add unique features as they attract more demand from customers. To meet this growing demand, the system design must be functional, efficient and attractive. Furthermore, by performing functional tests, they ensure that the software performs exceptionally well and that no errors are detected during its execution.

For the automated ticketing system, we need to analyse the accumulated unique characteristic that will be useful to fully satisfy the customer demand. So for ticketing, what functions should be tested:

• The system must receive adequate service 24/7(24 hours a day, 7 days a week); check whether this function is working properly or not.

• Check that the touch screen monitors and keyboard interface are working properly or have a problem.

• Test the touch screen menu to see if it works properly or not.

• On the touch screens, the train information on the screen, the arrival and departure time of the train, the ticket price and the reservation are displayed correctly or not.

• Customer can buy multiple tickets and the system will decide how many tickets a customer can buy at one time and as a tester we need to test this function to see if it works well or not.

• The ticket confirmation function that is only provided by the administrator. If the administrator wants to cancel the confirmation, the administrator can cancel the confirmation. Therefore, this feature is largely designed for the administrator and this feature should be tested properly.

• The system always updates the ticket availability information and you must verify the information correctly if it is updated or not.

• The customer can purchase their tickets with a credit / debit card. The payment method function works correctly or not and needs to be tested.

**Features not to be tested**

Automated Ticket Issuing System for Dhaka Subway Systems, there are some features that are not as much as important to test. The following features will not to be tested for our System-

* Customer’s all the information will store in that data base. But we are not going to test the information like customer’s name ,address, age, contact number , email address, job status etc.
* We don’t need to test the hardware functionality for Automated Ticket Issuing system.
* In our system though network testing is necessary, here we are not capable to do the network test.

**Approach**

Software testing is the process of analysing software to find the difference between the required and existing condition. Software testing is done throughout the software development life cycle and also to create quality software, for this purpose two basic testing approaches are used which are white box testing and black box testing. . We plan specification-based testing approaches by developing a wide variety of implementation-based testing techniques to be applicable to formal specification languages.

The Following section will be about Testing Levels:

**Unit testing**:

Unit testing is a method of software testing by which individual units of source code (sets of one or more software modules with associated control data, usage procedures, and operating procedures) are tested to determine whether they are suitable for an application. White-box testing method is used in Unit testing and it is generally done by the programmer.

In automated ticket issuing system project, we have to ensure 24/7 support for that we need almost bug free software with proper functionality checking before releasing our software. In our features we have complex requirements like multiple ticket issuing, .So, we need to make stable and scalable software and ensure all the requirements are fulfilled with proper functioning application.That’s why we need unit testing to ensure bug free, stable and proper functioning software.

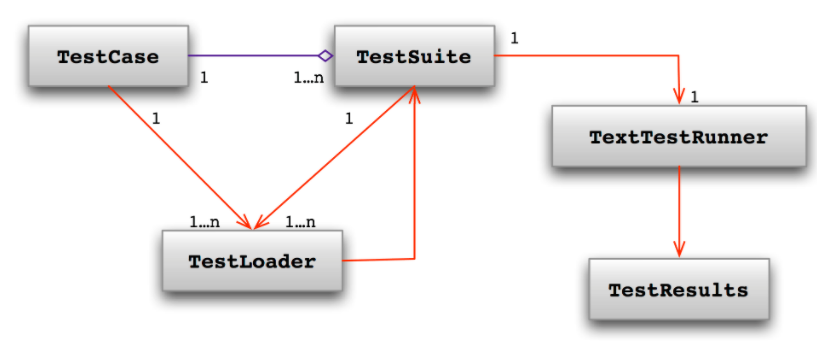


Fig: Unit testing

**Integration Testing:**

Integration testing advances one level above unit testing and examines how units integrate with each other. The main goal of integration testing is to detect interface errors.

Grey box test method (mixture of white box test method and black box test method) is used in integration testing, this test can be performed by programmers and testers. As Integration testing is the next logical step in unit testing and is part of the software testing process, this is where two testing tools come in: stubs (dummy lines of code written to simulate higher functions) and drivers (functions to call other functions).

In integration testing, we can combine the **Automated Ticket Issuing system** modules that we get from unit testing and we can test them as a group to make sure the system is ready for system testing. This test will also verify the data flow from one module to another module.

In our project, we want to ensure that one customer can buy one ticket .If any customer want to buy multiple tickets, the system should do the work. In this case we use integration testing because this testing use for condition checking method.

After buying tickets, customer have pay bill for tickets. Whenthe billing process completed thecustomer get the ticket. This data flowing process also observe in integration testing.

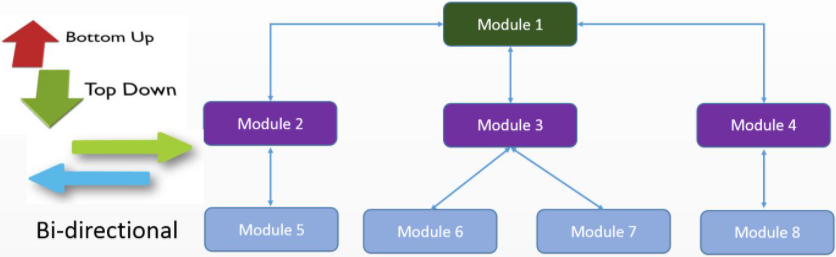


Fig: Integration testing

**System Testing:**

When we get to the system tests, we should have completed the unit and integration tests and have the software fully loaded in a test environment. The main purpose of system testing is to evaluate

The system meets the specified requirements. The black box test method is used in system testing. System testing should be done by a professional tester.

In our automated ticketing system, we focus on the following:

• **Usability testing**: the user interfaces of the system are easy to use and easy to understand or not.

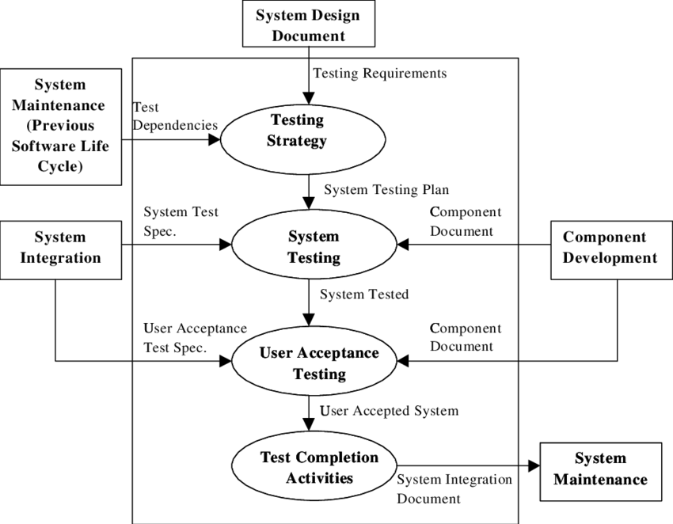
• **Documentation test** - When the customer uses the system, the system guide explains the functions as they work in real life. We will check it.

• **Functionality test**: as described in the documentation, the system behaves as expected or not.

**• Interoperability test**: Check that the system works with different operating systems, browsers, plugins, etc. Properly or not.

**• Performance Test** - Performance Test is the most important test for our automated ticketing system. We need to verify that our system can meet the requirements for speed, responsiveness, and stability under load. And it will help ensure a more positive user experience.

* **Security testing**: Our System can check testing validation (the input and output like **User’s Name,User’s Password**) or not.



**Fig: system testing**

**Acceptance Testing:**

When unit, integration, and system testing is complete, it’s time to accept the work has done all we can but likely only brushed the tip of the iceberg in terms of the sheer amount of massive errors we have made.

There are some bugs only users can catch because of the near-infinite combinations of units and use cases. That’s why user [acceptance testing](http://softwaretestingfundamentals.com/acceptance-testing/) is important.

In our system Acceptance test must be done by the End-Users and maybe done by the Admin to check the system is working properly or not. To measure our **Automated Ticket Issuing system** whether the system met all the requirements or not.

When the unit, integration and testing of the system is complete, it's time to accept that the job has done its best, but it has probably only touched the tip of the iceberg in terms sheer amount of massive mistakes we made.

If there are a few bugs that only users can catch due to the almost endless combinations of units and use cases, this is why user acceptance testing is important.

In our system, the acceptance test has to be done by the end users and maybe by the administrator to check whether the system is working properly or not. Measure our automatic ticketing system whether the system met all the requirements or not.

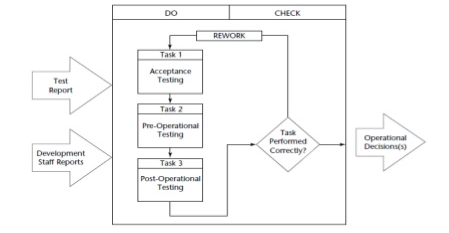


Fig: Acceptance testing

**Automated testing:**

Unlike manual testing, where humans must check each use case by hand and record the result, automated tests rely on tools to get the job done. For example, frameworks like Selenium will test web application compatibility between browsers. This is something that anyone who has tried to display their application correctly in Internet Explorer will know the value.

**Selenium**:

Selenium (Automated Cross-Browser Regression Testing) is free and open source. It has many high-end spin-offs, such as Sauce Labs, that rely on code to provide additional functionality. The tools created in this context are essential for testing browser compatibility.

**Postman**:

Tests are automated by creating test suites that can be run over and over again. Postman can be used to automate many types of tests, including unit tests, functional tests, integration tests, end-to-end tests, regression tests, mock tests, and more. Automated testing prevents human error and streamlines testing.

**CloudBees**:

Fully automated test suite from code to production. CloudBees includes unit testing and continuous integration features, and it gets started for free. Used by Netflix.

**CircleCI:**

CircleCI(continuous integration platform. Used by Facebook and Spotify) provides a comprehensive platform for automated testing, among other things, as its own build environment and deployment tools.

Thus, automated testing frees our system development team from boring weeks of testing after one role is created and motivates them to start working on the next one instead of worrying about the previous job. It is more efficient, offers better quality software, and reduces costs. Think of it in the same way as setting up automation. Setting up automation takes time, but saves much more in the future.

The automated ticketing system will be a more efficient and error-free application by automated testing. With the help of automated testing, we can easily detect errors, verify data flow and properly control flow tests, give an idea of how it looks on the web platform. We can also complete our project within budget and on time. Automated tests provide a visual representation of a project.

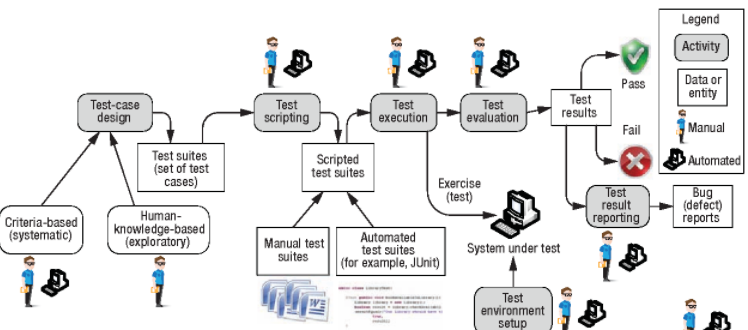


Fig: Automated Testing

**Item Pass / Fail criteria**

This section is for Pass and Fail criteria. Pass and Fail criteria is mandatory in the Test Design specification. As we know when the system requirement matches the all requirementthe test is considered as pass and when failure to meet requirement the test is considered as fail.

Here when components matches with the dedicated interfaces design then it will considered as pass. In testing time ifcomponent failure to meet the objectives of the object design specification, it will fail. Here Integration criteria pass when the interface environment team give a positive result after test and it is failed when failure to meet the objectives of both the object design specification and the system architecture specification. A system test pass or fail declaredafterdoing functional requirements, non-functional requirementsand use cases as an oracle test. A defect/issue will be reported in the defect tracking system for review by the triage team.

**Suspension Criteria and Resumption Requirements**

**Suspension Criteria**is a suspending the complete or part of the testing activities where**resumption criteria** imply resuming the previously suspended activities.

Some possible reason for suspension of the test:

* Unavailability of external dependent systems during execution.
* When a defect is introduced that cannot allow any further testing.
* Critical path deadline is missed so that the client will not accept delivery even if all testing is completed.
* A specific holiday shuts down both development and testing.

And the test will be resumes whenmeet the requirement again:

* When the external dependent systems become available again.
* When a fix is successfully implemented and the Testing Team is notified to continue testing.
* The contract is renegotiated with the client to extend delivery.
* The holiday period ends.

**Task Deliverable**

AIUB IT has responsibility for the following software testing deliverables and milestones

Phase 1 Testing Deliverables:

|  |  |
| --- | --- |
| Items Name | Date |
| Master Test Plan | January 2021 |
| System Test Results Document | February 2021 |
| Acceptance Test Results Document | February 2021 |

Phase 2 Testing Deliverables:

|  |  |
| --- | --- |
| Items Name | Date |
| System Test Results Document | February 2021 |
| Acceptance Test Results Document | February 2021 |

**Remaining test tasks**

* 3rd party and off-The-Shelf components.
* Infrastructure components.
* Website and GPS device of software interaction.
* GUI response and directly database testing.
* Create Acceptance Test Plan
* Create System/Integration Test Plan
* Define Unit Test rules and Procedures
* Define Turnover procedures for each level
* Verify prototypes of Screens
* Verify prototypes of Reports

**Environmental needs**

One separate, controlled system will be required for the initial phase of testing, setup as per one standard, complete office environment. In order to maintain the integrity of the test environment his network will not be accessible to anybody outside this project. The printers are also exclusively for use by the test network.

**Hardware components required**

* 1 Network Controller
* 6 Networked PC's (See below)
* 1 DAP Workstation
* 1 Oracle Server
* 1 HP LaserJet 4v Printer

**PC Specifications**

* The 6 PC's required for the test environment will include the following:
* 1 x P100, 1 GB SSD, and 16Mb RAM [Current Minimum Specification]
* 3 x P166, 1.5 GB SSD, and 32 Mb RAM [Current Standard Specification]
* 1 x P333, 2.5 GB SSD, and 64 Mb RAM [Current Maximum Specification]

These specifications are the various specifications currently in use in different branches. 1 x Pentium running Windows NT is also required as the Test centre for controlling and executing the automated testing.

**Softwarerequired**

**Test Tool:** Selenium, postman for automated testing

**Server:** Oracle database server, MySQL Workbench, Install MySQL server and Apache server

**System:** Windows 10 professional with latest Microsoft 365 access.

**Network:** Setup a LAN gigabit and 1 internet line with a least 10 MB/s

**Roles and Responsibilities:**

The following table represents the important staffs who will be responsible for the whole testing period and what will be their responsibilities:

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Member | Tasks | Staff number |
| 1 | Test Manager | Manage the whole project and define project directions with appropriate resources and guidelines. | 1 |
| 2 | Tester | * Identifying and describing appropriate test techniques/tools * Verify and asses the test approach and automation architecture * Execute test cases, log results and report the defects to manager. | 2 |
| 3 | Developer in Test | Implement the test cases (specifically which can be done by the developers e.g. Unit Tests). | 2 |
| 4 | Test administrator | * Builds up and ensures the environment * Assets are managed and maintained * Support tester to use the test environment for test execution |  |
| 5 | SQA Engineer | * Take in charge of quality assurance and create quality gates to ensure project quality and standard according to feature and budget. * Check to confirm whether the testing process is meeting specified requirements. | 4 |

**Schedule and Estimation:**

**All project task and estimation**

For estimation, we assume that our workforce will work 8 hours a day and 5 days in a week and according to that it will be a 40-hour work week and in total we will need 8 weeks 4 days.

|  |  |  |
| --- | --- | --- |
| Task | Members | Estimate effort |
| Analyze the software requirements | Test Manager, Tester | 20-man hour |
| Create test specification | Tester | 180-man hour |
| Execute test cases and Build Environment | Tester and Test Administrator | 100-man hour |
| Report the defects | SQA Engineer | 10-man hour |
| Monitor whole testing strategy | Test Manager | 40-man hour |
| Total |  | 350-man hour |

**Schedule to complete these tasks**

Following is the details timeline with the gantt chart of the project. We have created the timeline according to our estimation.

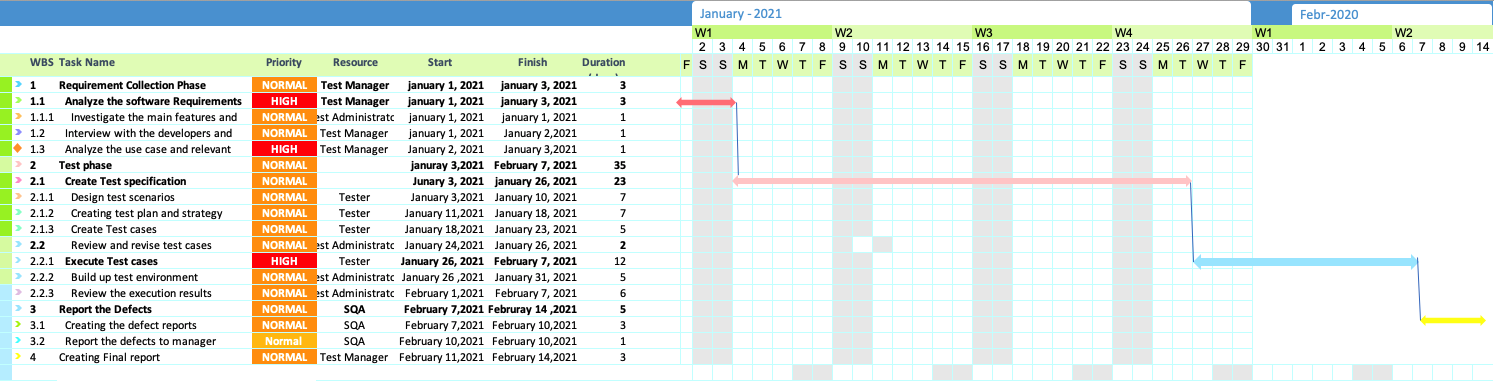


Figure: Full Test plan Timeline

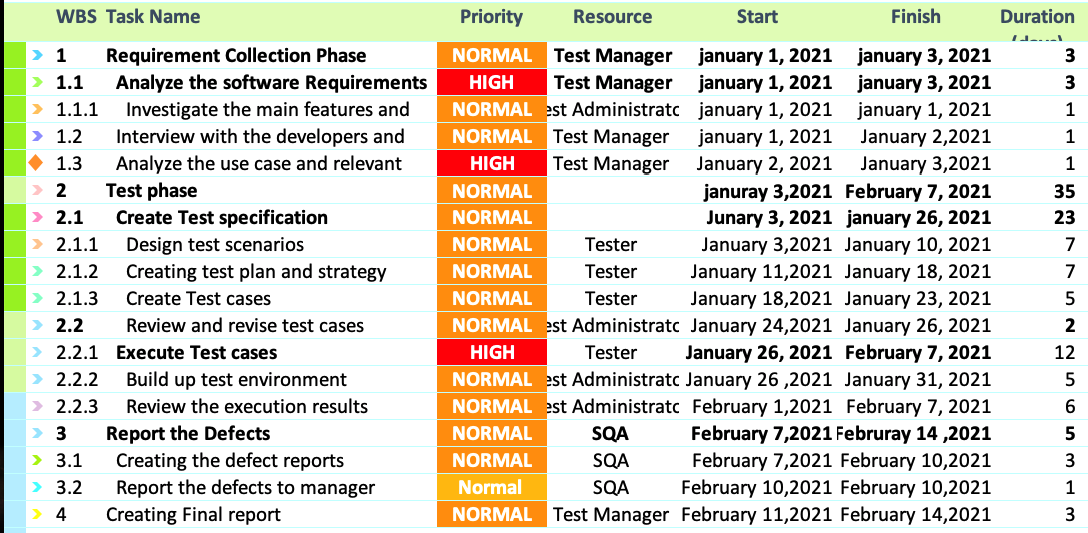


Figure: Test Plan Timeline

**Risk and Mitigation Plan:**

|  |  |
| --- | --- |
| Risk | Mitigation |
| Team members lack of the required skills for testing the complex functionalities | Plan training program to skill up and keep the test team updated. |
| The project schedule is too tight and hard to complete the project in time | Create the WBS first and set the test priority for each of the test activity. |
| Poor Management skill of the test manager | Plan leadership training for the manager |
| Wrong budget estimation and cost overruns | Creating details WBS and establish scope before starting the project and give proper attention in project planning |
| Leave important features untested | Scope analysis and analyze the requirements rigorously before starting the main testing period |
| A lack of cooperation negatively affects your employees’ productivity | Encourage each team member for doing their task and inspire them for a team effort. |

**Conclusion:**

The full test plan is based on developing the Automated Ticket issuing software for Dhaka Subway System. In the above plan the details of the procedure of testing the strategy we will follow is described including the task distribution and details timeline.