**Software Engineering**

**(IT-314)**

**EVENT PLANNER APPLICATION**

System Test Plan

**(version 1.0)**

**Team no: 14**

**Instructor: Prof. Asim Banerjee**

**Team Mentor: K Hemantha**

**Date: 23 February, 2016**

**Team Members**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Name** | **ID** |
| 1. | Patel Yash | 201301008 |
| 2. | Patel Tej Naginbhai | 201301018 |
| 3. | Anavadiya Yash Kumar | 201301052 |
| 4. | Chauhan Dwimitra Dharmendra | 201301079 |
| 5. | Rahul Shashank Saranjame (Team Leader) | 201301101 |
| 6. | Fatnani Varun Kumarbhai | 201301152 |
| 7. | Nikita Jain | 201301155 |
| 8. | Karri Kiran Satish Reddy | 201301168 |
| 9. | Raman Kumar Douchaniya | 201301202 |

**Version History**

|  |  |  |
| --- | --- | --- |
| **Date** | **Version no.** | **Author** |
| 23/02/2016 | 1.0 | Nikita Jain |

**Table of Contents**

1. Introduction ................................................................................................................. 5
   1. Purpose .............................................................................................................5
   2. Overview of Event Planner Application ...........................................................5
   3. Objective of System Test ..................................................................................6
   4. Formal Reviewing .............................................................................................6
      1. Formal Review Points............................................................................6

2 Test Items ......................................................................................................................7

3. Software Risk Issues .....................................................................................................7

4. Features to be tested ......................................................................................................7

5. Features not to be tested ................................................................................................9

6. Testing Approach .......................................................................................................... 9

7. Testing Strategy ............................................................................................................10

7.1 Types of Testing ................................................................................................10

8. Item Pass/Fail Criteria ..................................................................................................13

9. Test deliverables ...........................................................................................................13

10. Software and hardware needs .......................................................................................13

11. Responsibilities .............................................................................................................13

12. Schedule .......................................................................................................................14

13. Planning Risks .............................................................................................................14

14. References …………………………………………………………………………….15

**Overview:**

This document contains a set of planned activities to be conducted at the time of testing after the development of the modules has been completed. The system prepared after integrating all the separate modules that are intended to perform different functionalities will be tested. To ensure that the location for the presence of bugs is clearly identified, each and every module will also be tested separately. This document contains strategies to be followed while performing testing.

**Intended Audience:**

* Testing team
* Developers/Coders (for unit testing purposes)
* TA Mentor

**References:**

This document is prepared with the help of a previous few documents to develop a clear and thorough understanding which would facilitate creation of an efficient and realistic test plan. These documents are:

* Project Plan:
  + So that the test plan goes in accordance with the deadlines to be adhered to.
  + So that the test plan can be given achievable deadlines.
* Software Requirements Specification:
  + To clearly understand the requirements of the client.
  + To get acquainted with the system flow so as to get a thorough knowledge of the different system modules.

1. **Introduction:**

**1.1 Purpose:**

Testing is the process of validating and verifying the software product. Testing helps to contrast and critically analyse the differences, if any between the expected behaviour and actual behaviour of the software. Testing also helps to find out if the software system developed fulfills the requirements specified or not. The purpose of this document is to develop a test plan for our software application by establishing scope, strategies involved and schedule for the different testing activities as the software development proceeds. This test plan comprises of the description of the features of software system to be tested and the testing strategies to be employed on them. One of the important purpose is also to ensure that all the software modules perform their intentional functionality separately and also when integrated as a part of big system.

This document will describe the testing approaches to be used for testing the product. This will involve breaking one big software product into features to be tested and features not to be tested. This will be decided based on the requirements specified in the software requirements specification document. This document will schedule the testing activities along with the testing approaches to be used so that a fully functional and tested product can be developed within the available timeframe. There will always be possibilities of detection of software defects while testing, so a contingency plan has to be prepared for removing the detected bugs and make necessary changes in the modules or in interfacing the modules in the integrated system. The document will also enlist the resources required to perform testing activities and the criteria at which a test on any module or software system will be considered as either pass or fail.

**1.2 Overview of Event Planner Application:**

Event Planner is an application that we are making for our Ahmedabad based client All About Ads. The application is intended to ease the task of performing all the event planning activities for our client. The application will perform the planning activities such as bill generation, vendor selection, event notification etc. The application is loaded with the data specific to the client’s organisation. So, it will be specially for the in-house use of the client and will be used by the employees of the organisation.

**1.3 Objectives of System Test Plan:**

System test plan is intended to maintain the quality standards for the software product developed. One of the important objectives of system test plan is to ensure that the system meets the requirements in exactly the same manner in which they were specified. If bugs are founds, then the system test plan should include contingency time periods for the removal of detected bugs and development of fully functional and tested software product. System after being tested should satisfy the use case scenarios and should fulfill the standards established for considering a software module to be tested and verified.

System test plan also clearly specifies which features of the software product are to be tested and which are not to be tested. After testing is done, all the associated risks and software faults detected are to be informed to the associated development team members so that the a bug-free software product can be released within time constraints. It has to be extremely taken care of in the system test plan that the software release time does not fall ahead of the bug removal and changes incorporation period in the software modules. Thus, test plan has to be devised in such a way so that any changes in requirements specifications, functional requirements, non-functional requirements can be incorporated in the document form as well as in the actual software product. It should also be ensured that all such changes can be passed through highest level of testing again within the available time constraints for software product delivery.

**1.4 Formal Reviewing:**

A number of formal review points will be referred before and during the system test to maintain the performance of high quality testing activities. These review points include:

* Software Requirements Specification
* Design Documents
* Project Plan
* Unit test plans
* Unit test cases, conditions and results.
* System test conditions
* System test progress and results.
* Post test system review.

**2. Test Items:**

We intend to test following parts of the event planner application:

* The login feature is working as intended and the database is able to accept the entries of new users. The admin is able to accept and approve the new registrations or not.
* If the user is able to change password of his/her account or not.
* The user is able to retrieve the details for a specific event by using the event search functionality.
* The user is able to edit the vendor/event details in an event using different options provided and also if the user is able to save/remove the details after making the changes.
* Email notification feature is working as intended i.e, the user able to send email notification to the customer for any important change made in the event planning phase.
* The various save and submit buttons along with the field to make data entries are making the intended changes or not.
* If admin is able to delete an event if its details are not required anymore by the organisation. Deletion should remove the event from the database and should manage all the dependencies.

**3. Software Risk Issues:**

The potentials risks for the event planner software includes:

* If the system hangs at the client side
* If the system hangs at the server side or is unable to retrieve the response due to some reason.
* Modules independently do not produce the intended results
* After interfacing the modules to create the integrated system, results produced for different software functionalities are not as intended.
* A lot of changes were requested in the requirements which were incorporated in the modules but were not documented properly, for both change requests and modules .
* The changes requests introduced dependencies in the database which creates problems in deleting and editing the event details.

**4. Features to be tested:**

The important functionalities of the software application should be tested depending upon the user’s view and user’s expectations from them. These functionalities are represented in the form of modules prepared by critically analysing the requirements mentioned in the software requirements specification documents. These requirements are a form of user’s expectations from the software product to produce the intended output. The features of the event planner application that will be tested include:

* Login and Registration Module:
  + To accept the username and password and login to the respective account of the user.
  + To accept the details of a new user by using appropriate password and username given by admin.
  + For forgot password by taking e-mail or UserID, some random password will be generated and sent to corresponding e-mail.
  + After that user will be redirected to the change password page to change password using password sent by mail.
* Create Event Module:
  + To accept the details of the customer, event and redirect to the event details page for editing event details after taking some initial pitching details.
* Upload Event Gallery Module:
  + The user should be able to upload and retrieve the information like pictures, videos, presentations etc made during the pitching phase and post event phase of that particular event.
* Costing Sheet Module:
  + The cost sheet should be generated as according to the vendors selected, rates mentioned in the database, quantity entered by the user.
  + To be able to make adjustments in the vendor selection and acceptance whenever needed. The changes made should reflect whenever the event details are revisited.
  + User should be able to delete a previously saved details of a vendor from the costing sheet prepared so far.
  + User can also add new vendor which is not used before.
* Delete Event Module:
  + The admin should be able to delete the entire information of the events not required anymore in a single hit.
  + Deleting of information for a particular event should be independent of the details in other events.
  + The rights to delete the details of particular event should be only reserved with the admin. No other user should be able to completely delete an event.
* Email Notification Module:
  + The user should be able to notify the customer about the changes that are incorporated in the previous planning of his/her event. Also, the decision of whether notifying the customer or not for a particular change should solely depend on the user.
  + The user should get a feedback for the customer being notified successfully.

**5. Features not to be tested:**

There are a few features that are already provided by wordpress which we used for the front end development of our web application. We will not test those features. The other features that are not to be tested include:

* Platform dependency.
* Operating system dependency.
* Web browser’s compatibility to execute the front end features that are coded in Javascript, CSS, HTML.

**6. Testing Approach:**

First of all we will apply the approaches of unit testing to test the individual software modules so as to identify if the unit as a whole is functioning as it is intended to or not. This process will be carried out by the developers at the time of coding itself so that the faults detected in the individual unit can be removed then and there. It will be cost effective and more efficient to carry out unit testing during coding phase itself so that the possibilities of units not working fine at the time of integration can be ruled out. This way, we will only have to focus on interfacing errors between the software modules.

For testing our complete integrated software product, we will be following Bottom-Up approach of integration testing. In bottom-up integration testing, the entire system is broken down into a tree having different software modules as the nodes of the tree and the module dependencies as the branches of the tree. Bottom-up integration testing approach is then applied from the leaf node of the tree till the root node of the tree. In this testing approach, each and every node of the tree (i.e. software module) is first tested individually with the help of stubs and drivers to fulfill the required dependencies on other untested nodes. The next module in the hierarchy then tested is the unit whose all the subordinate modules (the modules it calls) have been tested. After a module has been tested, its driver is replaced by an actual module in the hierarchy i.e the next module to be tested and its driver.

We will use bottom-up integration testing approach because it reduces the overhead work of creating more stubs and drivers as compared to top-down integration testing. This way, the tester can focus more on testing and bugs detection in the individual modules and on the integrated system. This will also help testers in completing the testing process faster as the stubs and drivers created that will be created in top-down approach, will be the actual tested modules in bottom-up approach.

After completing the integration testing, we will move to validation testing where we will check whether we have developed the product right or not. This will include validating if all the requirements specified by the client are met in exactly the same way or not.

**7. Test Strategy:**

**7.1 Types of Testing:**

**7.1.1 Unit Testing:**

Unit testing is performed on the system modules/units to ensure that each unit performs its intended tasks by producing the required outputs. The purpose of unit testing is to ensure that the program logic produces the output as expected from that specific software functionality. Unit testing will be performed at the time of development by the developers without making the use of results from other modules. Unit testing is an important type of testing to be carried out before integration testing so that the error types that should be rectified at the time of integration testing are prominently interfacing errors.

To perform unit testing, all the use-case flows, data flows, condition flows and branch flows, sequence flows, activity flows are executed using valid and invalid data input to validate the following:

* The expected output and the actual output produced for the valid data input. If the actual output is not equal to the expected output, then error is recorded.
* The expected output and the actual output for the errors or warning messages displayed when invalid data input is used. If the expected error messages are not found equal to actual error messages displayed, then an error for those values is recorded.

**7.1.2 Integration Testing:**

Integration testing is performed to test entire system which is formed by combining all the software modules in a single software system. The main purpose of integration testing is to find out any interfacing errors that may occur while combining two or more modules as part of a single software system. Integration testing uses top down approach or bottom up approach for testing the system which is broken down into a tree having different modules as it nodes. After testing the modules independently, they are also tested after being integrated to make sure that interfacing has not affected the working of any particular unit. Passing the integration testing is an important factor to find out if the product is ready for deployment or not. This will be done by a group of testers in our team.

**7.1.3) Validation Testing:**

Validation testing refers to testing if the product that has been developed is according to requirements of the client or not. Verification refers to if we are building the system right i.e. we are complying with the requirements or not. Validation refers to if the system being built is right or not, i.e. if we have understood the requirements clearly and built the system according to it or not. Validation can also be used to test if the product fulfills its intended use when deployed on appropriate environment. This will be done by checking with the requirements stated in Software Requirements Specification document.

**7.1.4 Regression Testing:**

Regression testing is a type of testing to find out if a part of software system functionalities that have been previously tested still work correctly after introducing any kind of changes independent of the previous system or interfacing the system with other software systems. The success of this testing will give an assurance to the team members of the fact that older programming still works fine with the addition of new changes. This is usually done at the time of integration testing.

**7.1.5 System Level Testing:**

System testing is performed to check if the system meets all the requirements of the client stated in the software requirements specification document. The software product developed should meet both functional and non-functional design requirements of the system. To execute the real life working of the software product, it is done on a scenario much similar to production server where the software will actually be deployed. The test cases are then made according to end user perspective to record the defects where the software doesn’t produce the expected output for the user.

**7.1.6 Performance Testing:**

Performance testing ensures the responsiveness of the software application. For this testing, team members will check each and every feature of the system to ensure that the application doesn’t take longer time than expected to produce an output for any functionality. The delays can possibly occur due to frequent database accesses. Since our software is highly database intensive, it is very important for our software to pass performance testing. This will be done by the group of testers to check the time lags occurring for getting output for various system features.

**7.1.8 Acceptance Testing:**

Acceptance testing is the final level of testing performed accompanied with the client side to check if the completed product is as according to the requirements stated in the SRS. The client will check if the system is in the state where it is ready to be accepted for its use by the end users. For this, the client may randomly select a feature of the software system and check if it produces the intended output or not. If the system passes this testing, the client will accept the product.

**7.1.9 Configuration and Compatibility Testing:**

This is performed to check if the system is compatible with different software and hardware environments. Our software is a web application which will be compatible with all web browsers, except for internet explorer. We don’t have any specific operating system requirements either. So we will not apply configuration and compatibility testing to test our system.

**8. Test Pass/Fail Criteria:**

A software item will pass or fail a test based on the following criteria:

* The expected output and the actual output for a set of valid test cases input applied on a system is same or not.
* The system produces the intended warnings or errors for a set of invalid test cases inputs.
* The execution of a feature to produce any output gets completed in the expected time or not.
* For any changes in the existing features or addition of new features, the system does not deviate from its previous outputs.
* Update, delete, execution takes as small time as possible for having a good user experience.

**9. Test Deliverables:**

After carrying out testing activities, following will the deliverables:

* Test Plan document
* Test cases document
* Test report document (different versions for the number of times re-testing has been done).

**10. Software and Hardware needs:**

Hardware requirements for testing our software product includes computing devices with active internet access to run the web application. Our application is a client server application running on local-host. We need Xampp and Wordpress installed in server’s computer. We need any browser (except Internet Explorer) installed in client’s computer. Server computer also needs internet connection.

**11. Responsibilities:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Role** | **Responsibility** |
| Nikita Jain | Tester | Test plan, Test cases, Test Report |
| Kiran Reddy | Tester | Test cases, Test report |
| Rahul Saranjame | Tester | Test cases, Test report. |

**12. Test Schedule:**

Following major testing activities will be performed in the given order as the product development reaches close to testing phase of software development:

1. Unit testing
2. Regression Testing
3. Integration Testing
4. Performance Testing
5. Acceptance Testing

**13. Planning Risks:**

Possible risks to the software project includes:

* Incorporating a lot of changes and improper documentation of them. There will be no record of these changes to refer to while carrying out proper testing activities.
* Lack of proper personnel resources at the time of starting the testing activities.
* Delays incurred in delivering the software modules.
* Development team not able to understand the requirements clearly to convert them into intended working module.
* Lack of coordination in the dependent areas of software development.
* Delay in completion of the assigned responsibility to a team member which is causing the improper functioning of other dependent activities.
* The number of test cases included may not be sufficient to test each and every functionality of the software product.
* All the coverage criteria for white box testing of the system, i.e in-depth testing of program logic are not attained. This may leave out a functionality that will not be used frequently by the user untested and hence undetected. This defect will never be detected until it is used after deployment.
* All the outputs obtained while black box testing are as according to their expected output not achieved.
* Change in the scope of project can be a serious risk.

**14. References**

* System Test Plan v1.0, Team 14, IT314 Software Engineering, Winter 2015-16, DA-IICT
* Project Plan v1.0, Team 14, IT314 Software Engineering, Winter 2015-16, DA-IICT
* SRS v2.0, Team 14, IT314 Software Engineering, Winter 2015-16, DA-IICT