Test Plan

C-TALK

Version 1.0

Author:
Bhoopendra Singh(201452020)
Sunny Sankhlecha(201452061)

November 5, 2016

Indian Institute of Information Technology Vadodara

Team members :

TEAM MEMBER	ID
Bhoopendra Singh	201452020
Shikhar Dhing	201452021
Venkata Sandeep	201452037
Anjali Kumari	201452042
Vipin Sahu	201452051
Prahlad	201452052
Sachin Jangid	201452060
Sunny Sankhlecha	201452061
Kenneth Tenny	201452066

Revision History

Version	Description	Date	Authors	Reviewers
1.0	Test Plan	07/11/2016	Sunny, Bhoopendra	Kenneth

Contents

1 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. It is not a test design specification, a collection of test cases or a set of test procedures; in fact, most of our test plans do not address that level of detail. 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.

2 Intended Audience

- Testing team
- Developers/Coders(for unit testing purposes)
- TA Mentors

3 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 Requirement 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.

4 Introduction

- 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.

- Overwiew Of C-TALK Apllication:
  There is a huge communication gap between students and professor. We are making this application to bridge this gap, by making a question and answer application where any student can ask question and get solution to his problem by professor or students. It is often seen that most students are shy to ask question in front of the class. So we are providing such students a platform to freely and anonymously ask

question(s).

- Objective 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.

- 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.

5 Test Items

We intend to test following parts of the C-TALK 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 ask question.
- The user is able to give answers.
- Upvote/downvote functionality is working or not.
- The various submit buttons along with the field to make data entries are making the intended changes or not.
- If admin is able to recognize and deactivate acount of that user who are using obscence language.

6 Application Risk Issues

The potentials risks for the C-TALK application includes:

- If the system hangs at the user 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.

7 Features to be tested

The important functionalities of the software application should be tested depending upon the users view and users 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 users expectations from the software product to produce the intended output. The features of the C-TALK that will be tested include:

- Login and Registration Module:
  - To accept the college e-mail id and password and login to the respective account of the user.
  - To accept the details of a new user by using appropriate password given by admin via e-mail id.
  - For forgot password by taking e-mail id, some random password will be generated and sent to corresponding e-mail.
  - After that user will be redirected to the login page.
- Question Module:
  - The user should be able to ask the question after he/she logged-in.
  - The Feed of the question page must be prioritised with respect to time.
- Answer Module:
  - The user should be able to answer of a question.
  - The user should able to upvote/downvote of a particular answer.
  - The professor should be able to verify the particular answer if he/she finds the answer is upto the mark.
  - The Feed of the answers page consisting of the answer list of a particular question.

- Feed Page Module:

  - The feed of the user would be sorting according as per the topic of interest, time of addition of answer.
  - The generation of new feed should be based upon number of upvote/downvote, verified badge given by professor.

- Profile Module:
  The user should be able to change his/her Topic of interest, able to change password.

8 Features not to be tested

Features that are not to be tested include:

- Platform dependency.
- Web browsers compatibility to execute the front end features that are coded in Javascript, CSS, HTML.

9 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.

10 Test Strategy

10.1 Types of Testing

10.1.1 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.

10.1.2 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.

10.1.3 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.

10.1.4 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.

10.1.5 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 doesnt produce the expected output for the user.

10.1.6 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 doesnt 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.

10.1.7 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.

10.1.8 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 dont have any specific operating system requirements either. So we will not apply configuration and compatibility testing to test our system.

11 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.
- Upvote ,downvote, add question execution takes as small time as possible for having a good user experience.

12 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 retesting has been done).

13 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 and web. We node,npm,dependency module(installed by npm), installed in servers computer. We arealso using a cloud base database hosted on mlab.com. We need any modern browser (IE8 and above) installed in clients computer. Server computer also needs internet connection.

14 Responsibility

Name	Role	Responsibility
Bhoopendra Singh, Sunny Sankhlecha	Tester	Test plan
Kenneth Tenny	Tester	Test Cases
Shikhar Dhing	Tester	Test Report

15 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:

- Unit testing
- Regression Testing
- Integration Testing
- Performance Testing
- Acceptance Testing

16 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.

References

[1] Project Plan Ver1.0, Team IT-2 *IT303 Software Engineering.* , Autumn 2016-2017, IIIT-V

[2] SRS Ver2.0, Team IT-2 *IT303 Software Engineering*, , Autumn 2016-2017, IIIT-V

[3] http://istqbexamcertification.com/