Zoey Lee

# Term Project: *TalkyTalky*

Test Plan Document

## Table of Contents

# 1 Introduction

The introduction of the Test Plan document provides an overview of the entire document with purpose, scope, target audience, terms, test plan description, unit testing, and integration testing for the Chat Application. The aim of this document is to outline the test strategy and overall test approach for **TalkyTalky**. Also, it provides visible confirmation to stakeholders that adequate consideration has been given to the governing the test effort and to have them approve the strategy.

## 1.1 Purpose and Scope

The purpose of this document is to identify items that should be targeted by the tests and to outline the testing approach that will be used. Also, the required resources are identified and an estimate of the test efforts is provided in this document.
The testing scope shall be clearly addressed in the master and/or level test plan for that test level. The application shall be tested on the different platforms as described in the Requirement document. Also, a set of critical functions described in the Requirement document shall be tested to determine that the code will work on all platforms. This Test Plan will cover Unit Test and Integration Test as identified in the testing strategy.

## 1.2 Target Audience

The target audience for this Test Plan document includes the developer, individual users and stakeholders. The developer performs tasks specified in this document, and provides input and recommendations on the document. Moreover, the developer should be able to utilize this document to understand the scope of work that must be accomplished.

## 1.3 Terms and Definitions

| Term | Definition |
|---|---|
| Talker/Use/Client | Any person who is using the application and is communicating with others over Internet in real time. |
| Server | A program that provides connections between users and delivers |

| | |
|---|---|
| | messages from one user to others. |
| Stakeholder | Any person with an interest in the project including developers. |
| Use Case | A list of actions or event steps, typically defining the interactions between a role and a system, to achieve a goal. |
| Socket | One endpoint of a two-way communication link between two programs running on the network. |
| Port | A connection point or interface between a computer and an external or internal device. |
| TCP/IP | The basic communication language or protocol of the Internet. It can be used as a communications protocol in a private network. |
| GUI | Graphic User Interface. Type of user interface that allows users to interact with electronic devices. |
| Scenario | Detailed description of a use case, including rules, exceptions, boundaries, limits, etc. |
| Test Case | A specific set of test data along with expected results for a particular test objective. |
| Test Design | Describes how a feature of function shall be tested. |
| Test Plan/Test strategy | Describes the scope, approach, resources and schedule for the testing activities of TalkyTalky. This includes defining what will be tested, who will perform testing, how testing will be managed, and the associated risks and contingencies. |

# 2 Test Plan Description

This section is concerned with features to be tested/not to be tested, testing schedule, and release criteria of TalkyTalky.

## 2.1 Scope of Testing

Features to be tested:

- Create new account: Verify that the system allows creating new account with unique username and password for each user. Also, verify that the system accepts indefinite number of chat clients.
- Login/Logout: Verify that the system allows users to login to TalkyTalky only if their information exists in the user list. Also, verify that the system allows online users to logout by clicking “logout” button.
- Manage messages: Verify that the system allows users to send a message to a particular user and/or all online users. Also, verify that the system allows users to access their own message history sorted by dates.
- Notification when a user logs in/out: Verify that the system notifies users whenever a user logs in/out.
- Differentiating online/offline users: Verify that the system allows users to distinguish who is online/offline. Also, verify that the system allows users to send a message only to the online users.
- Working on different platforms: Verify that the system shall be successfully working on different browsers/platforms.
- Graphical User Interface: Verify that the system shall allow users to interact with computers through graphical icons and visual indicators.

Features not to be tested:

- Data structure for user list: A hash table, which is provided by Java, will be used to collect the list of users. Not test how data is stored and how the data structure is implemented.

## 2.2 Testing Schedule

| Milestone | Deliverables | Duration | Start Date | End Date |
|---|---|---|---|---|
| Design test case | Zoey Lee | 2 days | 05/20/17 | 05/21/17 |
| Write and execute registration | Zoey Lee | 5 days | 05/21/17 | 05/26/17 |
| Write and execute Login/Logout | Zoey Lee | 5 days | 05/21/17 | 05/26/17 |
| Write and execute send messages to all online users | Zoey Lee | 20 days | 05/21/17 | 06/07/17 |
| Write and execute send messages to a particular user | Zoey Lee | 20 days | 05/21/17 | 06/07/17 |
| Write and execute history retrieving | Zoey Lee | 9 days | 05/28/17 | 06/07/17 |
| Write and execute notifications for users' login/out | Zoey Lee | 5 days | 05/21/17 | 05/26/17 |
| Write and execute differentiating between online/offline users | Zoey Lee | 9 days | 05/28/17 | 06/07/17 |
| Unit testing | Zoey Lee | 7 days | 06/01/17 | 06/07/17 |
| Project report | Zoey Lee | 1 day | 06/08/17 | 06/08/17 |

## 2.3 release Criteria

- The code must compile and build for all platforms.
- The code must report 0 errors and warnings.
- The system will be tested to accept at least 10 users at a time.
- All planned tests run, at least 90% pass.
- Ready to release by June 8.

# 3 Unit Testing

Unit testing is performed during construction phase by TalkyTalky developer. The test is the initial testing where individual methods of the software are tested and the purpose of it is to validate that each method of the software performs as designed.

## 3.1 Registration

All users are required to register for TalkyTalky before being able to communicate with other users. A user enters a username and password to register, and the inserted data is compared with data in the hash table correcting Talkers of the application and that is stored into the table only if the inserted data doesn't exist in it. Before releasing the application, I will test out to register with different combinations of username and password at least 10 times and see if the outcome satisfies my expectation. The system must accept a new user only if they entered a unique username and non-empty password.

### 3.1.1 Register with a unique username and non-empty password

| Input | Expected output |
|---|---|
| Unique username and non-empty password | Registration must be successful and the inserted data is stored into the list of users. Then, the user will receive a message that registration has been successful and be sent to login page. |

### 3.1.2 Register with an existing username and password

| Input | Expected output |
|---|---|
| Existing username and password | Registration must be failed. The user will receive an error message to prompt registering again and the system will not allow the user to login until the user finds a unique username. |

### 3.1.3 Register with an empty username and/or empty password

| Input | Expected output |
|---|---|
| Empty username and/or empty password | Registration must be failed. The user will receive an error message to prompt registering again and the system will not allow the user to login until the user finds a unique username. |

## 3.2 Login/Logout

Once a user registered to TalkyTalky, they are able to login to the system with their own username and corresponding password. Only if a user enters a username existing in the list of users and a password matching with the username is accepted to login to the system. Before releasing the application, I will test out to login with different combinations of username and password at least 10 times and see if the outcome satisfies my expectation. The system must accept a user only if they entered a valid username and matching password.

### 3.2.1 Login with a valid username and matching password

| Input | Expected output |
|---|---|
| Valid username and matching password | Login must be successful. The user will receive a message that login has been successful and be sent to the main page. |

### 3.2.2 Login with a valid username and not matching password

| Input | Expected output |
|---|---|
| Valid username and not matching password | Login must be failed. The user will receive an error message to prompt login again and the system will not allow the user to login until the user enters valid username and corresponding password. |

### 3.2.3 Login with an invalid username and invalid password

| Input | Expected output |
|---|---|
| Invalid username and invalid password | Login must be failed. The user will receive an error message to prompt login again and the system will not allow the user to login until the user enters valid username and corresponding password. |

### 3.2.4 Login with an empty username and/or empty password

| Input | Expected output |
|---|---|
| Empty username and/or empty password | Login must be failed. The user will receive an error message to prompt login again and the system will not allow the user to login until the user enters valid username and corresponding password. |

### 3.2.5 Logout when successfully logged in

| Input | Expected output |
|---|---|
| Logout when successfully logged in | Logout must be successful. The user will receive a message that logout has been successful and be sent to login page. |

### 3.2.6 Logout when login failed

| Input | Expected output |
|---|---|
| Logout when login failed | Logout must be failed. The user is not able to see logout button nor click the button to logout successfully. Login is required before logout from the system. |

## 3.3 Manage messages

Once a user logs in to the system, they are able to send/receive messages and retrieve their own message histories. Communicating with other users is available only if both sender(s) and receiver(s) are online. Messages are shown only to the participants of the conversation and other users who didn't participate in the conversation cannot see the conversation. Before releasing the application, I will test out to message to different status of users at least 10 times and see if the outcome satisfies my expectation. The system must allow a user to message only if senders and receivers are online.

### 3.3.1 Message to all online users

| Input | Expected output |
|---|---|
| Message to all online users | Sending/receiving a message to/from all online users must be successful. The user can see all conversation that they are participated in. |

### 3.3.2 Message to a particular online user

| Input | Expected output |
|---|---|
| Message to a particular online user | Sending/receiving a message to/from a particular online user must be successful. The user can see all conversation that they are participated in. |

### 3.3.3 Message to an offline user

Even if a user logs in successfully, sending/receiving a message to/from an offline user must be failed.

| Input | Expected output |
|---|---|
| Message to an offline user | Sending/receiving a message to/from an offline user must be failed. The system will not allow a user to send a message to offline users. |

### 3.3.4 Access to own message history

| Input | Expected output |
|---|---|
| Access to own message history | Accessing to own message history must be successful by clicking history button. The history will be displayed sorted by time and dates and the messages will be what the user sent/received. |

### 3.3.5 Access to other user's message history

| Input | Expected output |
|---|---|
| Access to other user's message history | Accessing to own message history must be failed. The message history will be accessible only to the user who owns it. If the user hasn't participated into the conversation, they are not able to see the message. |

## 3.4 Notification login/logout

Once a user logs in to the system, they will receive a notification from the server whenever other users log in and out. Notification will be sent only to the current online users. Before releasing the application, I will test out to see if the server notifies to different status of users at least 10 times and if the outcome satisfies my expectation. The system must send a notification to the online users only.

### 3.4.1 Verify all online users receive a notification whenever others log in/out

| Input | Expected output |
|---|---|
| Verify all online users receive a notification whenever others log in/out | Receiving a notification by a server must be successful if a user logs in successfully. The server must notify all online users whenever other users log in/out. |

### 3.4.2 Verify all offline users do not receive a notification whenever others log in/out

| Input | Expected output |
|---|---|
| Verify all offline users do not receive a notification whenever others log in/out | Receiving a notification by a server must be failed if a user failed logging in. The server must notify all online users whenever other users log in/out. |

## 3.5 Differentiating online/offline users

Once a user logs in to the system, they will be able to see the list of online user on the side of the screen. Differentiating online/offline users is available only when a user logs into the system. Before releasing the application, I will test out to see with different status of users if they can see whose status is online/offline at least 10 times and if the outcome satisfies my expectation. The system must allow only online users to distinguish whose status is online.

### 3.5.1 Verify all online users can see the list of online users

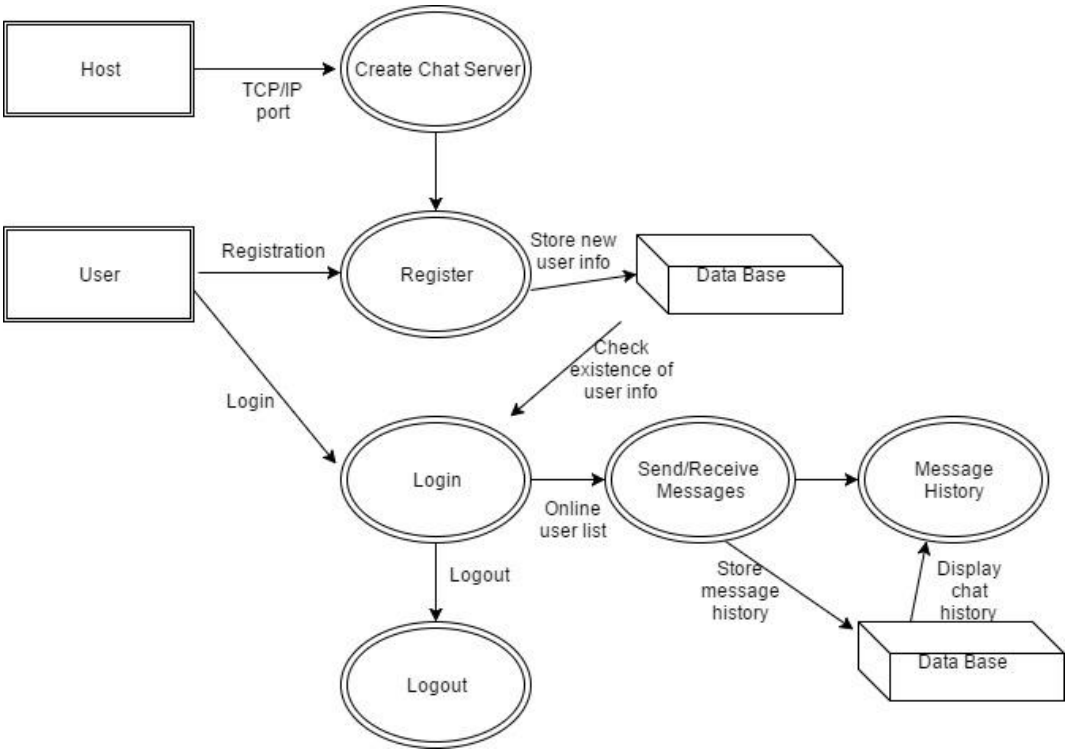
| Input | Expected output |
|---|---|
| Verify all offline users can see the list of line users. | Differentiating online/offline users must be successful if a user logs in successfully. The system must display a list of online users on all online users' screen and it must be updated every time a user logs in/out. |

### 3.5.2 Verify all offline users cannot see the list of online users

| Input | Expected output |
|---|---|
| Verify all offline users cannot see the list of line users. | Differentiating online/offline users must be failed if a user failed logging in. The system must display a list of online users on all online users' screen and it must be updated every time a user logs in/out. |

# 4 Integration Testing

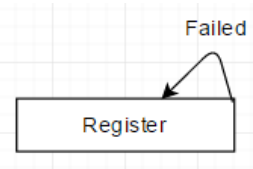
Integration testing confirms that individual unit of the system interacts as designed and that all functionality is working. The purpose of the testing is to expose faults in the interaction between integrated units. Figure 1 describes the data flow diagram of the entire system. Figure 2 describes possible testing scenarios for integration testing.




**[figure 1] Data flow diagram of the entire system**

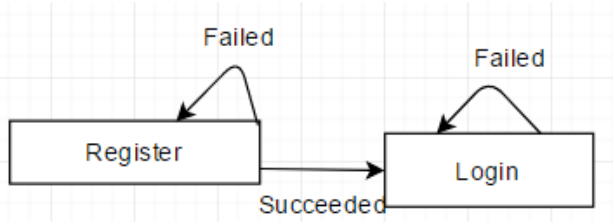
## 4.1 Register failed

If a user enters a username that already exists in the list of users or an empty username/password, the system will ask the user to choose different username and password. The user will be sent back to the registration page to prompt a unique username and non-empty password.



## 4.2 Register – Login failed

If a user succeeded registering for TalkyTalky, they will receive a message that registration has been successful and be sent to the login page. If the user enters an invalid username with password or a valid username with not corresponding password, the user will receive an error message and be sent back to the login page to prompt their own username and corresponding password.




## 4.3 Register – Login – Message to all online user - Logout

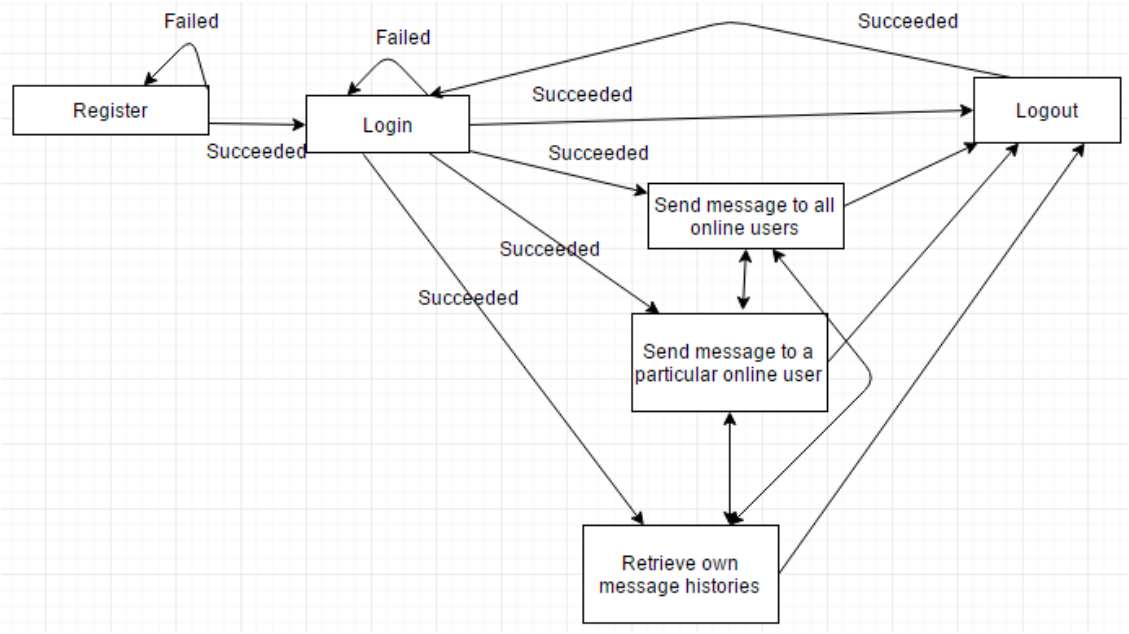
If a user succeeded registering and logging into TalkyTalky, the user will be sent to the main page where they can have a conversation with all online users. The list of online users is displayed on the side of the screen and is visible while the user is online. Whenever another user logs in/out, the user will receive a notification by the server and the list of online users is updated. The user can send/receive a message to/from all online users and the conversation with them is stored into all conversation participants' message history. By clicking logout button, the user is able to log out the system and is sent back to login page.

## 4.4 Register – Login – Message to a particular user - Logout

If a user succeeded registering and logging into TalkyTalky, the user will be sent to the main page where they can have a conversation with all online users. The list of online users is displayed on the side of the screen and is visible while the user is online. Whenever another user logs in/out, the user will receive a notification by the server and the list of online users is updated. By double clicking one of the usernames on the list of online users, the user can message to the particular user and the conversation with them is stored into their and the particular user's message history. By clicking logout button, the user is able to log out the system and is sent back to login page.

## 4.5 Register – Login – Retrieve message histories - Logout

If a user succeeded registering and logging into TalkyTalky, the user will be sent to the main page where they can have a conversation with all online users. The list of online users is displayed on the side of the screen and is visible while the user is online. Whenever another user logs in/out, the user will receive a notification by the server and the list of online users is updated. By clicking history button, the user will be sent to the message history page to retrieve their own message history. All displayed messages are what they sent or received and they are sorted by time and dates. By clicking logout button, the user is able to log out the system and is sent back to login page.




**[figure 2] Possible test scenarios for integration testing**