BatChat

A brief overview of the program’s

usage and design

Compiled By:

Gary Baldanza

Dalina Beckham

Kenneth Cassine

Welcome to BatChat!

The ultimate messenger designed to be used by Batman and all his affiliates, BatChat will serve as the next stage in instant messaging software for superheroes. BatChat will have the ability to network multiple users together for quick and convenient messaging, in addition to providing easily accessible public message feeds. This document will provide information concerning the usage of BatChat, in addition to how the program is funded, who the intended users are, the usage of the program, and how it is designed and implemented.

Document Purpose

BatChat may prove difficult to understand for users who are unfamiliar with how twitter, facebook, and other online forum or chat rooms work. This is especially true of users with exceptionally busy lifestyles, such as trying to save the world. This document will cover key concepts including program use cases, client information, and assumptions made during development. Use cases will include the different functions available for the different user categories, and how they relate to each other in the use case diagram.

System Scope

BatChat is an instant messaging system designed for easy functionality by both superheroes like Batman, and also for easy access by public officials such as the police. The chat program will have both private and “public” profiles available, ensuring that information is only provided to the source that needs it the most. Registered users will be able to send messages to other users with either private or public visibility, in addition to posting photos.

Content Overview

The rest of this document will contain details concerning the client for which BatChat is being made, the intended users of the program, and details about the actual program itself. This information includes use cases, the overall design of the program, and assumptions made during the development of BatChat. The next page details the remainder of the document as a Table of Contents.

Table of contents

|  |  |
| --- | --- |
| Section Title | Page |
| Overall Description | 4 |
| -Client Details | 4 |
| -The Users | 4 |
| -Product Functions | 4 |
| -Use Cases | 5 |
| -Use Case Diagram | 13 |
| System Specifications | 13 |
| -Class Diagram | 15 |
| Assumptions | 16 |
| In Conclusion | 16 |
|  |  |

Overall Description

Client Details

Through private funds acquired as a donation from Bruce Wayne, Batman has opted for the creation of this system so as to remove any dependence on other companies for communication, including Wayne Enterprises. This is meant to add a level of discretion to the chat system, in addition to providing for easier program customization due to direct client feedback. Information concerning Batman himself is classified and will not be provided in this document.

The Users

The users of BatChat will include both Batman himself and all of his affiliates. This includes but is not limited to fellow superheroes, carefully selected allies, and potentially law enforcement officials. Law enforcement officials will have access to the system as public users, allowing them to see messages that may be worth spreading by radio. Batman’s affiliates will be able to register with BatChat so that they are able to focus on sending messages both privately and publicly. Batman himself or whoever he deems worthy will have administrator access to control that BatChat network. The functionalities of each user type are explained in detail below.

Product Functions

BatChat is meant to provide the most basic chat features as securely as possible. These features are designated based upon whether the user is public, logged in, or the administrator. With each respective level, the functions available to the user increase.

Public users are those without any login credentials, and will only have access to messages posted with public visibility. Public users will be able to register with BatChat as a new user. Upon successful registry, they will be able to log into the system and become a private user, granting access to the bulk of the program functions. All users will have the ability to quit the program from the main menu.

Upon successful login, users will automatically see the public message feed, in addition to any private messages from users to which they have a subscription. Logged in users will be able to create and post messages of their own, and can control if the message has private or public visibility. An additional function of posting messages will include the ability to attach photographs and reply to a specific user’s message using the at symbol (@). Logged in users will also be able to subscribe to other users by providing the corresponding username to the system. This will result in the ability to see private messages from the subscribed user. Added functionality to the subscription feature will include both the ability to view a list of subscribed users and also to remove them as needed.

Logged in users can create or join groups, and will be able to post statuses as a group by utilizing the Hashtag symbol (#). From here, logged in users can do a search by Hashtag, allowing the user to view only the messages that match the search criteria. To finish out basic utilities, logged in users will have the ability to log out of the system, which will reset their status to being public, and can also change their password.

The Administrator, in addition to all functions available to the logged in user, will have the ability to directly control all users within the BatChat program, in addition to all of the features available to a regular user. The Administrator is able to view a list of all users registered with BatChat. The Administrator can also remove users should he or she feel the need to do so. Administrator login credentials are provided with BatChat upon initial delivery.

Listed below are the detailed use cases in alphabetical order, followed by the use case diagram.

Change Password

Short Description: A logged in User can request to change his or her password.

Main Flow:

1. The user selects the change password option
2. The program prompts for the original password
3. The user provides the original password
4. The program prompts for the new password
5. The user provides the new password
6. The program prompts for the new password again
7. The user confirms the new password
8. The program prompts the user that the password has been changed successfully

Alternate Flow:

1. The user selects the change password option
2. The program prompts for the original password
3. The user provides the original password
4. The program denies the change due to an incorrect original password
5. The user selects the change password option
6. The program prompts for the original password
7. The user provides the original password
8. The program prompts for the new password
9. The user provides the new password
10. The program prompts for the new password again
11. The user confirms the new password
12. The program denies the change due to the new passwords not matching

Join Group

Short Description: A logged in user will be able to join a group when selecting the option. The user will enter the name of the group. The group will be created if it doesn’t already exist. Otherwise, they will be able to see the list of hashtag messages of the group.

Main Flow:

1. User selects ‘Join/Create a group’ option.
2. User enters group name
3. Group gets created and prompted

Alternate Flow:

1. User selects ‘Join/Create a group’ option.
2. User enters group name
3. User is able to see list of group messages If group already exist and prompted.

Log In

Short Description: After selecting the log in option, the user enters a username and password, after which they will be viewed by the program as a logged in user. If the credentials fail, the program will return to the main menu.

Main Flow:

1. User selects Log in option
2. Program prompts user for name and password
3. User enters username
4. User enters password
5. Program confirms success
6. User is returned to main menu as a logged in user

Alternate Flow:

1. User selects Log in option
2. Program prompts user for name and password
3. User enters username
4. User enters password
5. Program denies provided credentials
6. User is returned to main menu as a public user

Log out

Short Description: Logged in user selects the logged out option, upon which they become a logged out user.

Main Flow:

1. User selects the log out option
2. The program requests confirmation of selection
3. The user confirms selection
4. The program returns the user to the main menu as a public user

Alternate Flow:

1. User selects the log out option
2. The program requests confirmation of selection
3. The user cancels the log out
4. The program returns the user to the main menu as a still logged in user.

Post Messages

Short Description: A logged in user will be able to post messages when selecting the option. The user will be asked to post publicly, or to subscribers. The user will post the message, and the message will be updated in message file. The program will deny the message if it exceeds 140 characters.

Main Flow:

1. User selects ‘Post message’ option
2. User types in the message
3. User asked to post publicly or to subscribers
4. The program posts the message into the feed
5. The program updates the message file

Alternate Flow:

1. User selects ‘Post message’ option
2. User types in the message
3. The program denies the message because it exceeds 140 characters

Alternate Flow 2:

1. User selects ‘Post message’ option
2. User precedes message with the hashtag symbol and a word
3. User types in the message
4. User asked to post publicly or to subscribers
5. The program posts the message into the feed
6. The program updates the message file

Quit/Exit

Short Description: The user prompts the program to exit, upon which the program saves all data before prompting the user to confirm exit selection. The user will then confirm the exit selection and the program will close.

Main Flow:

1. User selects exit option
2. The program requests confirmation of the exit selection
3. The user confirms the exit selection
4. The program updates the user's file (it is possible that no changes are made)
5. The program updates the messages file (it is possible that no changes are made)
6. The program closes.

Alternate Flow:

1. User selects exit option
2. The program requests confirmation of the exit selection
3. The user cancels the exit selection
4. The program returns to the main menu

Register

Short Description: A public user registers credentials with the system so as to be able to log in.

Main Flow:

1. User selects the Register option
2. The program prompts the user for a username
3. The user provides the username
4. The program prompts the user for a password
5. The user provides the password
6. The program confirms successful registration

Alternate Flow:

1. User selects the Register option
2. The program prompts the user for a username
3. The user provides the username
4. The program denies the username because it is already taken
5. The user is prompted to use a different username
6. The user may either provide a different username or cancel

Reply

Short Description: The user replies to a message within his or her feed.

Main Flow:

1. The user selects the message to which he/she wants to reply
2. The user creates the message, preceding with the @username designation
3. The program attaches the reply to the message

Alternate Flow:

1. The user selects the message to which he/she wants to reply
2. The user creates the message, preceding with the @username designation
3. The program denies the message because it is greater than 140 characters
4. The program denies the message because it was not preceded with @username

Remove Subscription

Short Description: A public user selects the remove subscription option, upon which the program either confirms its removal or the removal fails due to an invalid username. The user may also cancel the remove subscription option.

Main Flow:

1. User selects the Remove Subscription option
2. The program prompts the user for a username
3. The user provides the username
4. The program prompts the user that the subscription has been removed
5. The program refreshes the view feed with the new settings

Alternate Flow:

1. User selects the Remove Subscription option
2. The program prompts the user for a username
3. The user provides the username
4. The program prompts the user the subscription could not be removed, as the user is not in the subscription list

Alternate Flow 2:

1. User selects the Remove Subscription option
2. The program prompts the user for a username
3. The user cancels the actions
4. The program returns to the main menu

Remove User

Short Description: The administrator may remove a user from the system. The action will cancel if either a false username or password is provided.

Main Flow:

1. The administrator selects the Remove User option
2. The system prompts the administrator for the username
3. The administrator provides the username
4. The system prompts the administrator for his or her password
5. The administrator provides the password
6. The user that matches the provided criteria is removed from the system

Alternate Flow:

1. The administrator selects the Remove User option
2. The system prompts the administrator for the username
3. The administrator provides the username
4. The system prompts the administrator for his or her password
5. The administrator provides the password
6. The system prompts the administrator that the removal failed due to either an incorrect username or password.

Search for Hashtag

Short Description: A new/logged user will be able to search for hashtag when selecting the option. The user will enter the word to be searched, if existed. Then the program will display the messages of hashtags.

Main Flow:

1. User selects ‘Search for Hashtag’
2. User enters the hashtag word
3. User then sees list of hashtag messages

Alternate Flow:

1. User selects ‘Search for Hashtag’
2. User enters the hashtag word
3. The user is prompted that the Hashtag doesn’t exist.

Subscribe

Short Description: A logged in user will be able to subscribe to another user when selecting the option. The user enters name of the user. The user then is able to see the list of posts from the subscribed user. The system will tell the user if the subscription attempt failed.

Main Flow:

1. User selects ‘Subscribe’ option
2. User enters name of subscriber
3. User then sees list of posts from the subscriber and is prompted

Alternate Flow:

1. User selects ‘Subscribe’ option
2. User enters name of subscriber
3. The system prompts the user that the subscriber does not exist

Upload Photo

Short Description: The logged in user uploads a photo to the message feed through a message. Just like posting a message, it will be denied if the text field has more than 140 characters.

Main Flow:

1. The user creates a new message
2. The user enters text into the message if desired
3. The user posts a photo attached to the message
4. The program posts the message

Alternate Flow:

1. The user creates a new message
2. The user enters text into the message if desired
3. The user posts a photo attached to the message
4. The program denies the message because the text exceeds 140 characters

View Feed as Public

Short Description: As a public user, can choose the view the public feed without logging in.

Main Flow:

1. The user selects the view feed option
2. The program displays the publicly viewable messages in the feed

Alternate Flow:

1. N/A

View Subscriptions

Short Description: The logged in user selects the View Subscriptions option, upon which the system will provide the list of subscriptions. From here, the user can return to the main menu.

Main Flow:

1. The user selects the View Subscriptions option
2. The program prints out the list of subscriptions
3. The user selects the return option
4. The program returns the user to the main menu

Alternate Flow:

1. N/A

View Users

Short Description: The administrator selects the View Users option from the main menu. The program prints the list in alphabetical order. The administrator can then return to the main menu.

Main Flow:

1. The administrator selects the View Users option
2. The program prints out the list of users in alphabetical order
3. The administrator selects the return option
4. The program returns the administrator to the main menu

Alternate Flow:

1. N/A

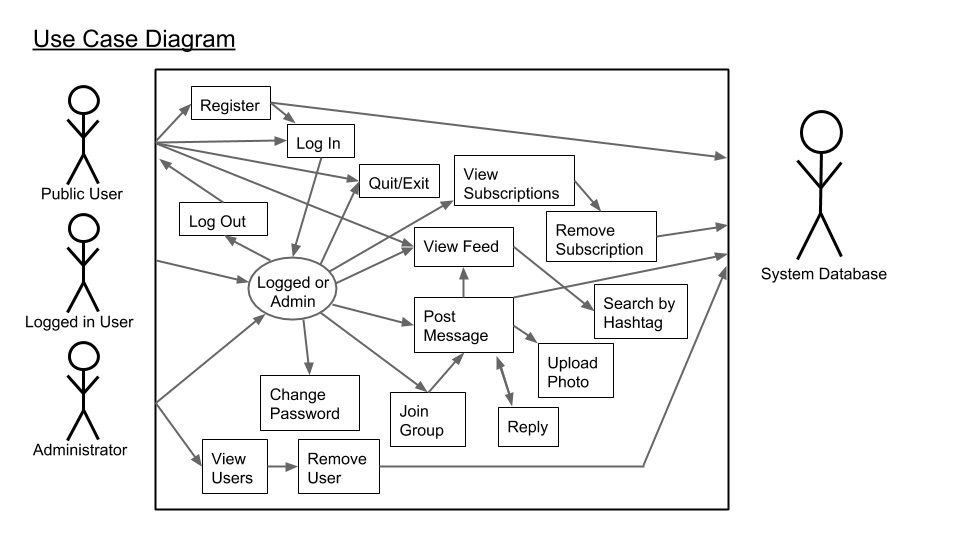


Image 1

System Specifications

BatChat is constructed around four basic sections, known as classes. These classes are the Interface, Users, Messages, and Hashtags. The Interface class will handle the primary methods through which the user will be able to interact with the program, while the other three classes are used to hold the data of the program. Each of the data classes will also have simple functions to aid in providing data throughout the rest of the program. The User class will hold the information unique to each user of BatChat, including the username and password. Each Message will store the actual message, who authored it, and any other Messages attached to it as a thread. The Hashtag class will keep track of both messages and users, in addition to a searchable keyword by which users will be able to find messages related to the Hashtag. A detailed explanation of this follows below, starting with the data classes so that the Interface will have greater clarity.

The User class is used for verification purposes in the Interface function, and is also used by both Hashtags and Messages to determine authorship of their respective content. Each User will hold both a username and a password, which will be used to verify provided login credentials. In addition, the User class will keep a running list of other users, which are defined as subscriptions. This is to allow a logged in user to have access to private messages from users to which the the logged in user has a subscription. The basic functions of the User class includes the ability to create new Users, known as the user constructor, the ability return information to other parts of the program, the ability to change a password, and the ability both add or remove subscriptions.

The Message class, much like the User class, is relatively straight forward. The Message class will be used to hold the posted message, in addition to knowing which user authored it, whether or not another message is attached to it as a thread or reply, a photo should the creator choose to upload one, and a status of whether or not the message is public. In addition to a message constructor, the Message class will be able to print the information it holds onto the screen, and can provide either the privacy setting or the author of the message. The ability to attach photos to the message will be a separate method within the class, and will not be part of the constructor.

Hashtags are used to aid in utilizing groups of users and the messages they create together. Because of this, Hashtags can keep track of both Messages and Users. Hashtags will also keep track of a keyword, which will be utilized when doing a search for a Hashtag. As an added feature, Hashtags will keep track of the total number of Hashtags that utilize an identical keyword, which will allow for monitoring how often that particular Hashtag is used. The functions utilized by the Hashtag will be limited to a simple constructor, the ability to return or display information, and the ability to add either users or messages.

The Interface class can be considered the main function of the program, and will hold the majority of functions available for the program. The Interface will hold an array of the hashtags, users, and messages, each of which will be loaded from their respective text files. If the text file for each class is not found, then it will start off as an empty array. The primary variables for the Interface class, excluding the actual arrays, will control how the interface is displayed to the user. These variables are the name of the current user if he or she is logged in, a variable determining whether there is a user logged in, and a variable determining if that user is an administrator. Each function of the interface will check the respective status of these variables to determine if the option is displayed.

The most basic function of the Interface is the ability to login and logout of the system. This will simply set the respective status variables to true or false, depending on the function, and will refresh the interface to reflect the new status. Public users, or users that are not logged into BatChat, will see the options to login, register, or view the public messages. Register will prompt the program to create a new user, upon which the user will provide a new username and password. The program will then determine if the username is taken by cycling through the current list of users and comparing the provided username to name of each user in the database. If no matches are found, the registration happened successfully. The function to view public messages will go through the array of messages and display all of the messages that have public visibility. This option is only available to the public user, as being logged in will automatically display all messages that are publicly visible, in addition to any messages to which the active user is subscribed to the message’s respective creator. As a final note, all users will have the ability to quit the system. This will tell the program to write the User, Message, and Hashtag arrays to their respective storage files before the program closes, which will allow for the data to be loaded upon reopening.

Upon logging into the system, most of the other features of the program become available. That being said, only a few of the uses actually rely on methods within the Interface itself, as other capabilities can be handled by other classes. An example of this is prompted the system to provide a list of subscriptions. While the option to do so is displayed by the Interface class, the actual labor within the program is handled by the User class. Those functions that are left include searching the message and hashtag database to provide information to the user, creating messages, joining groups, and special functions available to the administrator. Viewing messages will go through the message database, check for if it is public, and if it is not will compare the author of the message to the subscription list of the currently logged in user. For every match found, the message will be displayed. Searching by hashtag is very similar to viewing messages, except the program will instead search for each hashtag that matches the provided keyword, and will display all the messages attached to that keyword after passing a privacy check. Creating a message will call upon the message constructor, and add that message to the database. Joining a group will involve adding yourself to a particular hashtag collection. After providing the group credentials, the program will display a list of all groups that match the criteria. The user will then be able to choose from the list provided by the program, and be added to that particular hashtag. If the group does not already exist, a new hashtag will be created, and the user will be added to the newly created hashtag.

The administrator will have two capabilities that allow for added utility and functionality of BatChat. The first is the ability to view a list of all the users, which is handled by simply going through the current array of all users and displaying them. The second capability is actually removing a user from the system. To accomplish this, the program will search through the user database based on the provided credentials. If the particular user is found, it will be removed from the system.

Featured below is the official Class Diagram.

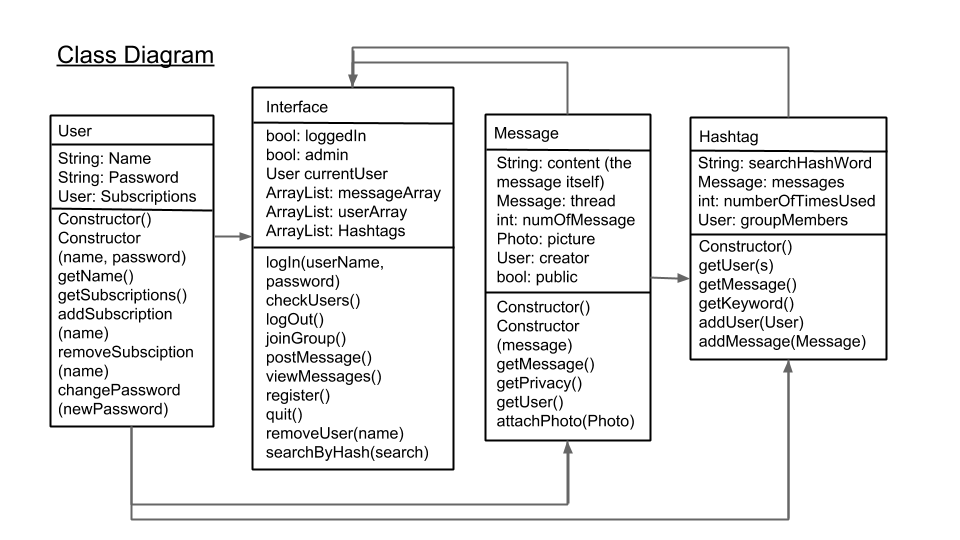


Image 2

Assumptions

Most of the assumptions made concerning the BatChat program are either security related or dependent upon the user. In each case, most of the problems presented can be mitigated by proper precautions or good practices taken by the users. BatChat also requires an understanding by the user that they should destroy their system if compromised so that BatChat doesn’t fall into the wrong hands. The assumptions and the decisions they resulted in are as follows:

* Users are expected to make strong passwords. BatChat will not check to make sure the user has made an adequately strong password
* Anyone who has access to this system will be able to register a new account. It will be up to the administrator to remove users who should not have access to the system.
  + We did not, in contrast, make the administrator confirm the creation of every account. While this method would indeed be much more secure, we assumed that few individuals would be able to access a system with BatChat installed.
* We are assuming that a system holding BatChat will not fall into the wrong hands. As a result, the data files in BatChat are not encrypted.
* The administrator’s credentials must be kept in a safe place by the owner of BatChat. We assume zero liability if these credentials are lost or forgotten.
* The users of BatChat are expected to have good conduct when posting pictures. BatChat does not have any form of censoring.
* The program currently only allows the existence of one administrator, which is built into the system. It does not allow the creation of new administrators (not even by the administrator).
* We have assumed that everyone who uses BatChat will immediately love it, and will never want to delete their own account.
* Whoever creates an account will not be able to change their username once it is chosen. The user must choose wisely, and not be overly disgruntled if the username they wanted is already taken.
* We would like to reiterate that BatChat does not have any form of censoring, and that good user conduct is expected in the creation of both messages and usernames.
* We assume that everyone using BatChat will have some other method to communicate with each other so as to safely exchange usernames without needing to post a public message.

In Conclusion

We hope you all make great use of BatChat, as we take great pride in knowing that this product has the opportunity to contribute in making the world a better place. Through your input, BatChat will become one of the premier tools in aiding the advancement of justice for everyone. We would like to extend our sincere gratitude to Bruce Wayne for his financial support. We would also like to thank Batman for his efforts against crime, and for inspiring us all to do our part in making the world a better place.