**Group N**

**Mini Twitter Project**

**Software Requirements Specification**

**For Front-end and Back-end**

**Version 1.0**

**Table of contents**

[2. Use Cases 1](#_Toc22969)

[3. ER Diagram 3](#_Toc7732)

[4. Detailed design: 6](#_Toc22891)

[5. System Screen 9](#_Toc7039)

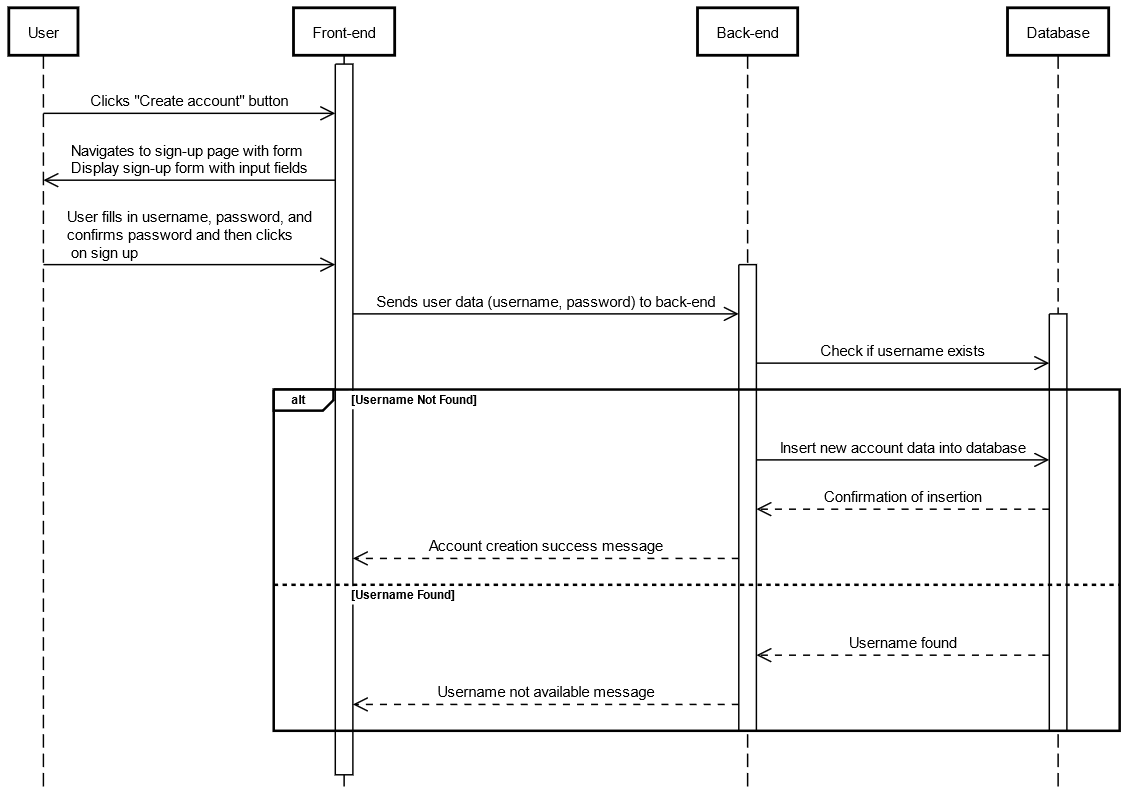
[6. Memos of group meetings and possible concerns of teamwork 13](#_Toc4362)

[7. Git Repository 14](#_Toc30793)

#### Use Cases

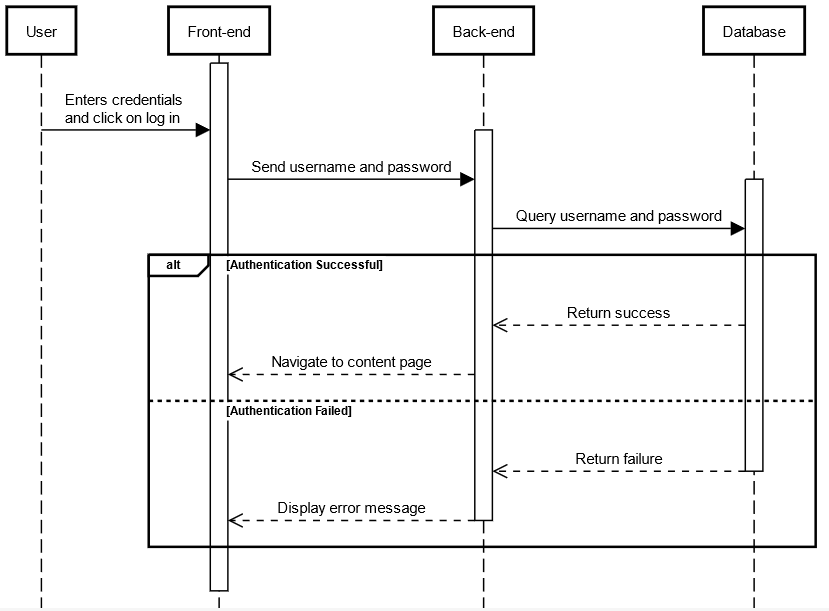
Scenarios 1：Create Account

user clicks on the “Create account” button on the sign-up page. The system navigates users to a new page with a form that is composed of three input fields, username, password, and confirm password. After users fill the fields with corresponding info and click the submit button, the front end will send the data to the back end. In the back-end, the system will first check whether the username exists in the database. If the username does not exist, the back-end system will input the new account’s data into the database and then show a message in the front end telling the user that the account has been created successfully.



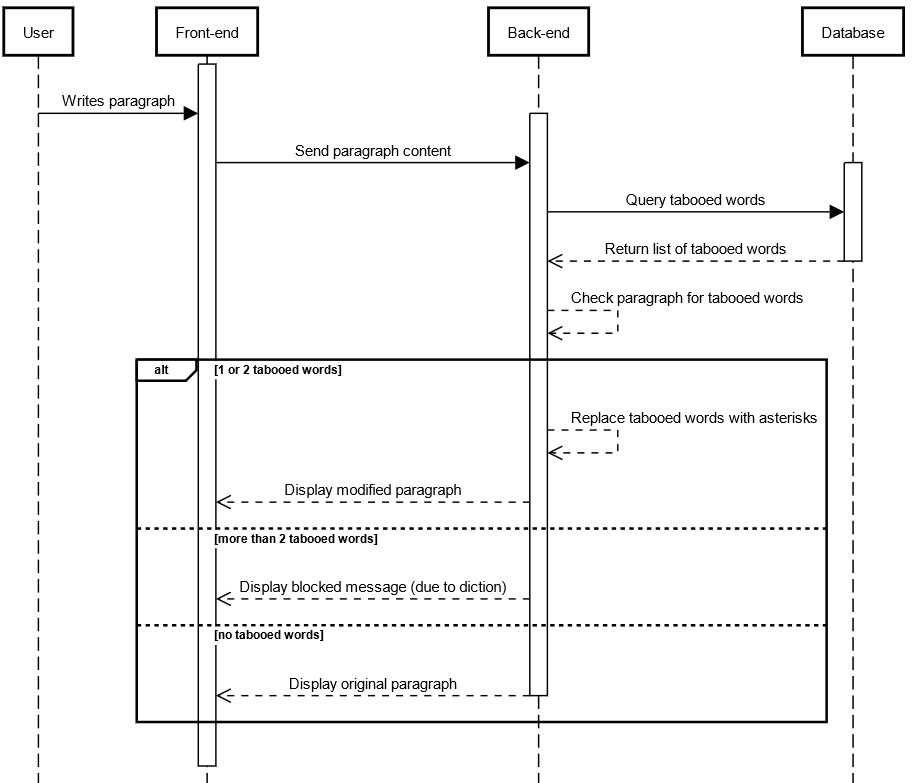
Scenarios 2：Log In

User stands in the log in page and input their username with corresponding password into the input fields and clicks on the log in button. The website which is the frond end of the system will then send the data of username and password to the back end of the system. The back end of the system will use query to search in the database if the given pair of username and password exists. Then the back end of the system will navigate user to context page if the authentication is processed successfully or display a message that username or password is not matched.



Scenarios 3：Post content

A common user had logged into the system and had wrote a paragraph in a content box. The user then click on the post button. The front end of the system will then send the content into the back end of the system, server. Server will use query to get a list of tabooed words from the database. The server will then check the context if it contains tabooed words. If there are 1 or 2 tabooed words, the server will change these words into number of asterisks then post to the web. If there are more than 2 tabooed words, the server will block the message and respond to the front end with a message that saying the context is blocked due to diction. Otherwise, the message will be post to the web.



#### 3. ER Diagram

**Primary Keys (PK):**

1. User: User ID

2. Message: MessageID

3. Keyword: KeywordID

4. Profile: ProfileID

5. Follow: A composite key (FollowerID, FollowedID) might be used.

6. TabooWord: TabooWordID

7. Click: ClickID

**Foreign Keys (FK):**

1. User: UserID may be referenced as a foreign key.

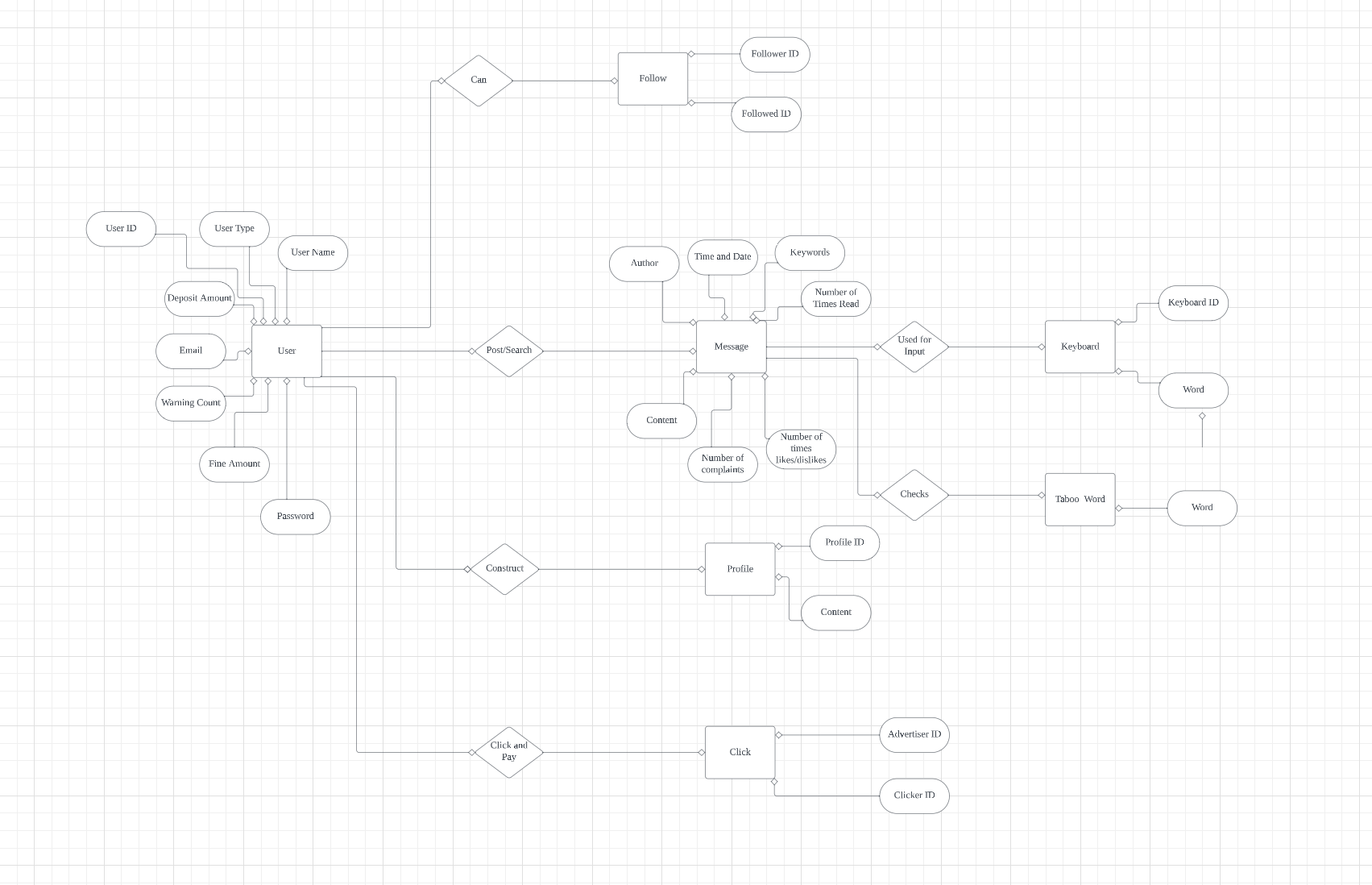
2. Message: AuthorID (referencing UserID in the User entity).

3. Profile: UserID.

4. Follow: FollowerID

5. TabooWord: No foreign key.

6. Click: AdvertiserID, ClickerID.



#### 4. Detailed design:

signUp()

//User types in their username and password in the input field and hits submit

//Make a post request to the server taking the username and password and storing it in the database

login()

//User types in their username and password in the input field and hits submit

//Make a get request from the server taking in username and password

//If the combination of the username and password is found

//Return the user's information

//If the combination of the username and password is not found

//Display an error message

searchMessages()

//User types in something in the search box

//System takes the input from the search and the variable, a json, that stored all of the messages

//If the input matches some words in the json, display it

getTop3MostLikedMessages()

//Make call to server to obtain all of the messages

//If call is successful, return a json of Message objects

//Iterate through the json and find the top 3 liked messages

//Return the top 3 liked messages

//If call is unsuccessful

//Return the error message

getTop3MostTrendyUser()

//Make a get request to server to obtain all of the users (return should be a json of user objects)

//If call is successful, return a json of user objects

//Iterate through the json and find the top 3 trendy user

//Return the top 3 liked user

//If call is unsuccessful

//Return the error message

suggestAccounts()

//Make get request to server to obtain all of the user's likedPost (return should be a json of post ID)

//Make get request to the server to obtain all of the authors of the IDs of the posts that the user likes

//Store the authors into an array

//Iterate through the array and find the top 3 common authors

//Return the top 3 common authors```

likePost()

//If user is not logged in (is a Surfer)

//If post is already liked

//Unlike the post

//Decrease the number of likes of that post by 1

//If post is not liked and the user disliked the post

//Undislike the post

//Like the post

//Decrease the number of dislikes of that psot by 1

//Increase the number of likes of that post by 1

//If the user is logged in

//If post is not liked

//Like the post

//Increase the number of likes of that post by 1

disLikePost()

//If user is not logged in (is a Surfer)

//If post is already disliked

//Undislike the post

//Decrease the number of dislikes of that post by 1

//If post is not disliked and the user liked the post

//Unlike the post

//Dislike the post

//Decrease the number of likes of that psot by 1

//Increase the number of dislikes of that post by 1

//If the user is logged in

//If post is not disliked

//Dislike the post

//Increase the number of dislikes of that post by 1

postMessage()

//User types in a message and submits it

//System takes that message and calculates how many words there are

//System checks the balance of the user

//If balance is greater or equal to than the message bill

//Deduct the balance from the bill

//Make a post request to the server and store the message in the database

//If the balance is less than the message bill

//Issue a warning and change the flag hasEnoughBalance in the database of that user to false so that when the user logs in again, they will be directed to the payment page

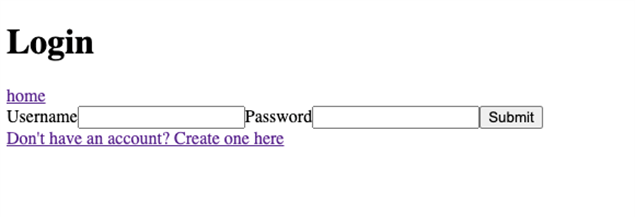
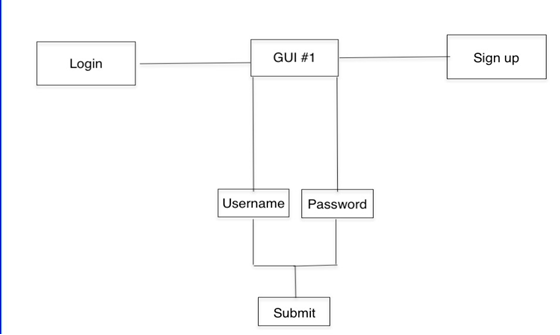
postComment()

//User types in a comment on a post and hits submit

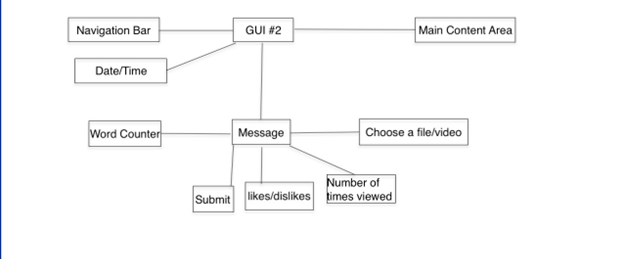
//Make a post request to the server to store the message as well as the ID of the person who wrote the message in the database

#### System Screen

There are two GUI screens in this project so far. When the user runs the program, it’ll take them to the first GUI screen which is the “Login” layout. The “Login” layout will ask for users to type their username and password. Once the user types of their account information, they will hit the submit button, which will take them to the next screen. If users run the program and don’t have an account, they need to create one by clicking the link “Don’t have an account, Create one here”. The next GUI screen includes graphics that will take the user after signing in. On the very top, we’ll include the navigation bar so that the user can have access to most of the components in the home page. This screen demonstrates a message box that users will use in case they need to upload something. This message box includes a word counter to check how many words the user types. The user could add several files and videos to their message. Once the message is posted, it’ll show up to the main content area. After that, the users will be able to like/dislike the message and view it several times. The following pictures show screenshots for the GUI layouts and diagrams. The layouts describe how the project looks like and the diagrams illustrate the components that each GUI contains.



GUI #1

GUI #2

#### Memos of group meetings and possible concerns of teamwork

# **Date: 10/7/2023 Saturday 3PM**

Topic: Project Overview

Preparations: Review project specifications given by professor

Discussion points

1. What language are we using? (Website? GUI?)

2. What are your personal skills (what are you comfortable with)?

3. Who is going to do what?

4. Setting up deadlines

**10/7/2023 Meeting Notes**

- Backend Team (Jinfeng & Xinwei)

- Frontend Team (Da Yuan & Dimo)

- We will be developing a website

- Backend will use SQL for our database system

- Frontend will use HTML, CSS, and Javascript with React as our JS Framework

- For now, frontend team will focus on developing the design of the website (not pretty) to develop all the features and interactions from the backend. Backend team will wait for frontend team to finish or partially finish it and they will take over and develop the database system

**11/13/2023 Meeting Notes**

\* There are several parts to the project: collaboration class diagram, ER diagram, diagrams for the use cases, pseudo code for every function, GUI demo, and sample function demo.

\* You can use the diagrams in iml-diagrams channel for help

\* Da Yuan Zhao will work on the pseudo code functions, Jinfeng will work on the diagrams for the use cases

\* Deadline to submit report #2 is 11/21 midnight. Submit your part by 11/20.

\* We need to get on our toes and accomplish our parts for the project because we are really really behind

\* Deadline for the project is on 12/11 which is less than a month

**According to Da Yuan Zhao, Team is really behind, and we are making effort to catch up.**

#### 7. Git Repository

https://github.com/dyzhao1011/mini-Twitter