SafeTweet, a Secure Social Network with Sensitive Information Detection

Yihan Liao

School of Computing Science

Sir Alwyn Williams Building

University of Glasgow

G12 8RZ

A dissertation presented in part fulfillment of the requirements of the Degree of Master of Science at the University of Glasgow

12th December 2021

Abstract

Social networking platforms have become an integral part of people's lives. Various social networking platforms serve different purposes. For instance, Facebook allows people to share their personal experiences with friends, relatives, and acquaintances, whereas people post messages to interact with a broader audience on Twitter. On the other hand, LinkedIn is an employment-oriented online service where people seek jobs and increase their visibility to potential employers. However, there is no professional social networking platform to share sensitive information about their job (e.g., workplace atmosphere, salary, benefits) while preserving their privacy. SafeTweet is a web-based platform where people can freely post workplace tips and seek advice on doing their jobs. Users can also learn about the benefits and salaries of other employees working in similar positions, and such transparency can facilitate fairness. Users looking for a job can learn about working environments and workload in various companies. In SafeTweet, information security is guaranteed for all users who want to post sensitive information but would not want their employers' or co-workers to see (e.g., low salaries, toxic working environments, lack of benefits). It allows users to send posts anonymously and encrypts them to block employees of the same company. In addition, SafeTweet also has a powerful, sensitive information detection feature, including sensitive information detection of post content and Named Entity Recognition (NER). The accuracy rate of sensitive information detection based on the workplace was 83.85%. When people are not confident about the content of their upcoming posts, they can use the system's detection results of sensitive information as a reference. The final test of the system is mainly divided into software tests and user evaluation. The purpose of software testing is to check whether the system’s core functions run normally. User evaluation aims to assess how potential users of the system perceive it during and after interacting with it. The user evaluation outcomes were later analysed to identify how SafeTweet should be improved in the future.

Education Use Consent

I hereby give my permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic form.

Name: Yihan Liao Signature:

Acknowledgements

Many thanks to Dr Handan Gül Calikl from the School of Computer Science at the University of Glasgow for providing a lot of help and guidance on all my questions and taking the valuable time to have an extra meeting when I needed support. Many thanks to the evaluators of this project for their valuable time to provide suggestions for my project. Thanks to Twitter and Postman for delivering the developer platform to help the project get a high-quality data set. I would like to thank my parents and friends for their encouragement and support during my project. Finally, I would like to thank the University of Glasgow for allowing me to study, which has profoundly impacted me.

Contents

Chapter 1 Introduction

This section describes the current potential demand for workplace social platforms (Section 1.1) and how SafeTweet meets this need (Section 2.2).

1.1 Background and Challenge

Social networks are an integral part of life for most people. People make friends on Facebook and Twitter, find jobs on LinkedIn. As evidence for people's heavy reliance on social media platforms, Twitter is the second-largest social network, has more than 1 billion registered users and 326 million active users [1]. They are very willing to share their daily life on those platforms. However, the existence of social networks inevitably forces people to face privacy issues [2, 3]. For professionals, except for the usual information leakage, also have the risk of leaking the content of tweets. Due to hundreds of millions of people using social networking platforms, some people may think that they will not be discovered when unscrupulous talking about sensitive information related to the company, such as salary, revealing the idea of job-hopping, complaining about colleagues, company and employer. However, the recommendation system of social networks would constantly recommend users to workmates or friends of friends [4]. So many ways can lead to leakage of improper comments. Employees who post sensitive information might face pressure from their companies if companies discover their inappropriate statements. Some professionals use non-public social accounts to send posts. But if a post includes any name, location, or company can also be easily recognised by people around them. Therefore, it is practical to post sensitive company information on social platforms without being detected.

In addition, people are increasingly suffering from employment pressure due to the COVID-19 pandemic, leaving many job seekers in a weaker position. From March to April 2020, the unemployment rate in America rose from 4.4% to 14.7% [5]. Not only because of the depressed market, lack of transparency between job seekers and companies is also a crucial reason. Companies always want to select fewer demanding employees, such as accepting lower pay or working overtime without complaint. Corporations can sift through resumes, compare candidates and choose the best value for money. Job seekers are stuck waiting again and again. Even if he does land a job, he may encounter a poor working environment but doesn't realise it until he starts working. There is also information inequality in salary, which job seekers are most interested in. Newly graduated students have no idea of the salary level of different cities, companies and different positions. So, they may suffer loss in the salary aspect when looking for a job. Besides, people seldom talk about salary. If someone gets a disproportionate amount of money, they may not find out. Some efforts have been made to address this situation. In 2016, LinkedIn launched Salary Insights, which gathers salary information from members to provide salary insights to job seekers [6]. In 2017, Kenthapadi proposed the LinkedIn compensation product, which helps people calculate their earning potential by collecting a large amount of data [7]. These are rewarding and innovative products. But at present, the coverage of these products is low and is not suitable for many regions and positions. Also, many job seekers are looking for

information other than salaries, such as working environment and intensity. Therefore, it is an urgent need for a secure social platform, which allows people to talk freely about their careers without the risk of being discovered.

1.2 Solution

SafeTweet is a web-based real-name social network based on the workplace. People mainly share workplace updates in this social network, regarded as Twitter based on the workplace. However, unlike other social networks, SafeTweet allows people to post anonymously, which means they can hide their information while posting without fear of being discovered by the boss. Since SafeTweet includes an anonymous feature that allows users to post anonymously. If the content of a post contains sensitive information and personal entity information, it can also help users be anonymous and encrypt post content. The encryption function is an encoding process that transcodes text content using Base64. Other employees cannot see all encrypted messages of the same company. When employees at other companies see the post, they can click the decrypt button next to it and navigate to the decrypt interface. The decrypted page contains the translated content of tweets, and copying is prohibited. The page is also full of watermarks with the reading user's real name, which helps prevent people from taking screenshots or photos to spread it. In addition, if users are not sure whether their text content contains sensitive information, they can use the system's sensitive information detection mechanism. If there is sensitive information in the text content, NER detection is continued, with the main detection objects being name, organisation, location and money. If the text contains sensitive information, but the NER tool does not detect identity, the system will prompt users to remain anonymous. If the system identifies both sensitive and entity information, it will suggest user encrypt it.

1.3 Structure

The following section mainly introduces sensitive information detection and the tools needed to develop the system. The third chapter contains collecting requirements and the design of module functions. Specific system architecture and design details are outlined in Section 4. Chapter five describes how to construct each part of the system and the accuracy of the text analysis models. System testing and user evaluation are in section 6. Finally, the future work and conclusions are discussed in Section 7.

Chapter 2 Related Work and Tools

This section covers related studies in the literature (Section 2.1) and the tools used to develop SafeTweet (Section 2.2).

2.1 Related Work

In 2011, Mao proposed three tweets that could leak privacy and are worthy of attention [8]. Vacation tweets, drunk tweets and illness tweets. He used Naïve Bayes and the Support Vector Machine (SVM) classifier to classify sensitive information. The experiments indicate that the Naïve Bayes performed better than SVM. The accuracy of holiday tweets is 76% in Naïve Bayes. But the range of sensitive information involved is relatively small in this study, with only three aspects. In 2014, Islam divided tweets into 200 topics [9]. The detection probability of privacy information is significantly increased by pre-defining the theme of the content published on social networks and detecting sensitive information according to the related features of the topic. The author also found that Naïve Bayes had a better performance than classification. Both above studies are limited to identifying leaks of sensitive information from published tweets. In the context of increasingly serious privacy issues and people's increasing attention to privacy, it reflects the importance of real-time monitoring of sensitive information. In 2017, Cappellari built a privacy decision tool to alert users of potential privacy disclosure risks before sensitive messages leak to social platforms [10]. He used five algorisms, including nearest neighbour, Naïve Bayes, SVM etc. In his study, the SVM obtained the highest accuracy. Besides, in the same year, Neerbeky developed a real-time privacy detection desktop application based on Recurrent Neural Net (RNN) [11]. However, the author does not provide specific data for model accuracy. In 2018, Canfora used Natural Language Processing (NLP) to detect sensitive information in social networks [12]. The method is to judge sensitive information by analysing sentence structure, word order and context, rather than relying on specific data sets. However, the problem brought by this method is the incompleteness of the heuristic set, making it easy to misjudge complex sentences.

Nowadays, social networks are all based on the Web or mobile end. If it is a privacy detection system for social platforms, embedding the detect system in the Web end would achieve high availability. This thesis will focus on the work scene to study the performance of sensitive information detection in social networks. The data of choice were tweets containing workplace keywords, including work, job, colleague, workmate, boss, salary, wage, overtime, and a host of other privacy terms. As the research direction of this thesis is highly targeted, the selection of keywords is small and accurate. For the same number of tweets, using keywords yields more tweets available.

2.2 Related Tools

VSCode: VSCode supports various programming languages, including JavaScript, TypeScript, CSS, and HTML. It can also download extensions for

Python, C/C++, Java, and Go and debug Node.js. Fully meet the language requirements of the project.

JavaScript: JavaScript is a function-first, lightweight, just-in-time compiled programming language, which on the Website controls the behaviour of the Web page [13]. JavaScript is one of three languages that Web developers must learn, including HTML and CSS. Most of all web pages today are developed based on JavaScript.

Vue.js: Vue is a lightweight JavaScript library developed by You in 2014. The features of this framework are data binding and components development. For those who have learned the basics of the front-end, the framework is easy to use and has good performance. It is more popular on Github than React and Angular [14]. For the lightweight development goal of this project, Vue is suitable to be used as a front-end framework.

Bootstrap: Bootstrap is a front-end page framework developed by Mark Otto and Jacob Thornton, designers of Twitter. It is based on HTML, CSS and JavaScript and written by the dynamic CSS language Less, which provides an elegant specification for the front-end. Bootstrap also has a framework for Vue, called Bootstrapvue, which help Vue developers use Bootstrap.

Node.js: Node.js is a suitable server-side for real-time applications and multiple front-end technologies such as Vue and React [15]. It uses an event-driven, non-blocking I/O model, making it lightweight and efficient [16]. Currently, versions 12.x and 14.x is still being updated and maintained, but 12.x has entered the more stable Maintenance stage. Therefore, we chose the most stable version of JavaScript as the runtime environment, 12.16.0.

npm: A Node.js package management and distribution tool that helps developers quickly install packages and dependencies needed for a project. Currently, the latest version is 8.1.2, and I choose to install 8.1.0.

Python: Python is used to train models, detect sensitive information and use NER tools. Python3 is a stable version that is not compatible with Python2.x [17].

MySQL: MySQL can run all medium and large databases, suitable for Web development. It can handle the database containing tens of millions of orders of magnitude records and different run-on systems. It is the most widely used relational database management system [18]. MySQL has the advantages of small size, fast speed and low cost. Besides, it is open-source, allowing most small and medium-sized websites to choose MySQL when choosing their database. MySQL is a very suitable database for this project.

Chapter 3 Requirements

This chapter introduces collect demand by interviews and selection of interviewees. After compiling all the obtained requirements into a list, prioritise them through the MoSCoW method. The specific requirement list is in appendix A.

Collect requirements through interviews at the beginning of the project. The interview aims to understand employees' behavioural characteristics and preferences using social platforms. All the questions should refine according to the interview purpose, and conservations should build around the work and social network. Users interviewed need to be screened in combination with SafeTweet features, namely, people who like to use social networks and are already working or about to work. Divide users into core users and potential users. Core users are those who want to be active on workplace social platforms, and potential users are defined as the target user but not currently considered using SafeTweet.

The core users of this research object are two employees of Internet companies and a Human Resource (HR) of an Internet company. They are both avid users of social networks and have 2-3 years of work experience. Potential users are two graduates who have multiple social network accounts but have little experience in the workplace. Before the interview, different questions should be chosen for different types of interviewees, and each interviewer should answer about ten questions. After the interview, collate all content for the requirements and prioritise functions using Moscow. Figure 1 is the classification of the Moscow method [19]. Figure 2 shows the number of MoSCoW for each functional module. The detailed requirement list is in appendix A.

Category	Criteria
Must	**Must** have requirement
Should	**Should** have if at all possible
Could	**Could** have but not Critical
Won't	Would be good to have (**Won't** have time to do it now, but maybe later)

Figure 1: MoSCoW Criteria.

Module	Must	Should	Could	Won't	Total
Basic Function	3	2	0	0	5
Send Post	7	0	1	0	8
Information Detection	3	0	0	0	3
Read Post	3	1	0	0	4
Decrypt Post	1	1	1	0	3
Comment Post	1	2	1	1	5
Search Post	1	2	0	0	3

Figure 2: MoSCoW Criteria.

Chapter 4 Design

According to the requirement list, the overall system structure is designed. Starting from the bottom layer, cover the Run environment (Sections 4.1), Database (Sections 4.2), Data layer (Sections 4.3), Function layer (Sections 4.4), and front-end UI (Sections 4.5). The system structure design is as follow.

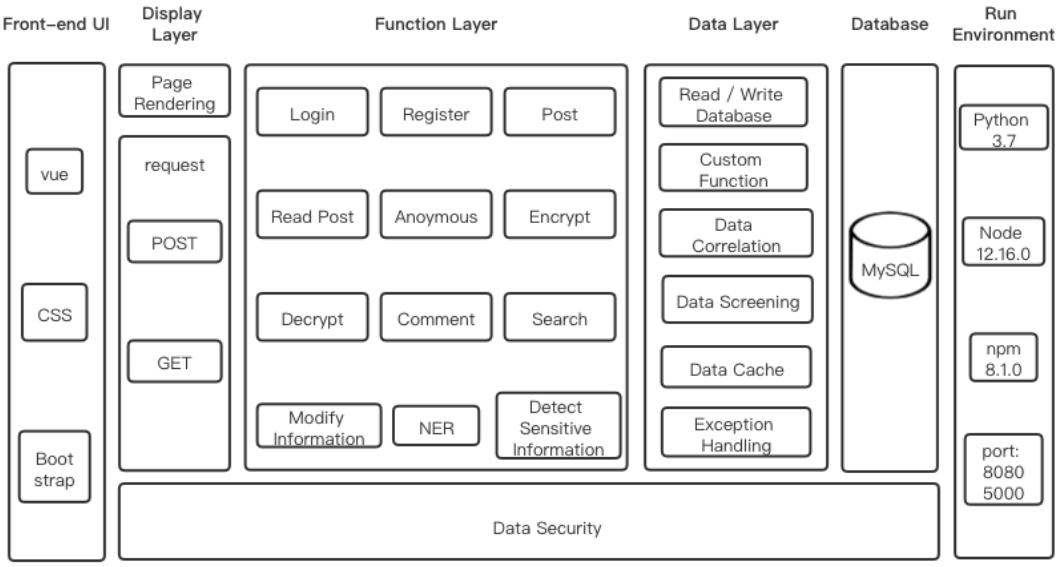


Figure 3: System Structure.

4.1 Run Environment

Port: Ports cannot be occupied by multiple services. The author selects 8080 and 5000, which does not conflict with the system, as the front-end and back-end port of the project.

4.2 Database

Based on the analysis of system requirements, the database tables involved in the system are designed, including users, tweets, comments and likes. The scheme of ER diagram is as follow.

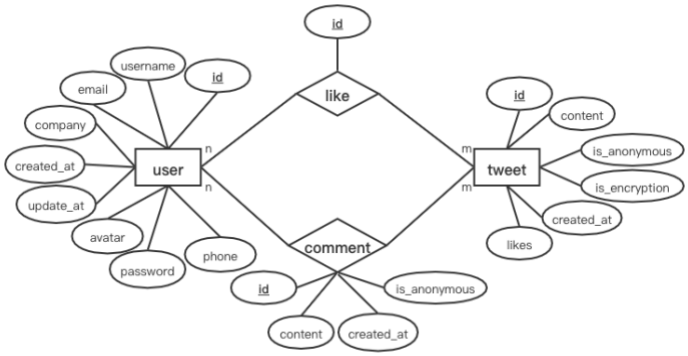


Figure 4: ER Diagram.

The relationship pattern of this project is:

1. **user** (**id**, username, email, company, avatar, password, phone, created_at, update_at)
2. **tweet** (**id**, content, anonymous, encrypted, created_at, like)
   Foreign key: user id, user name
3. **comment** (**id**, content, created_at, anonymous or not)
   Foreign key: user id, tweet id
4. **like** (**id**)
   Foreign key: user id, tweet id

The overall design and foreign key relationship of the database are shown below.

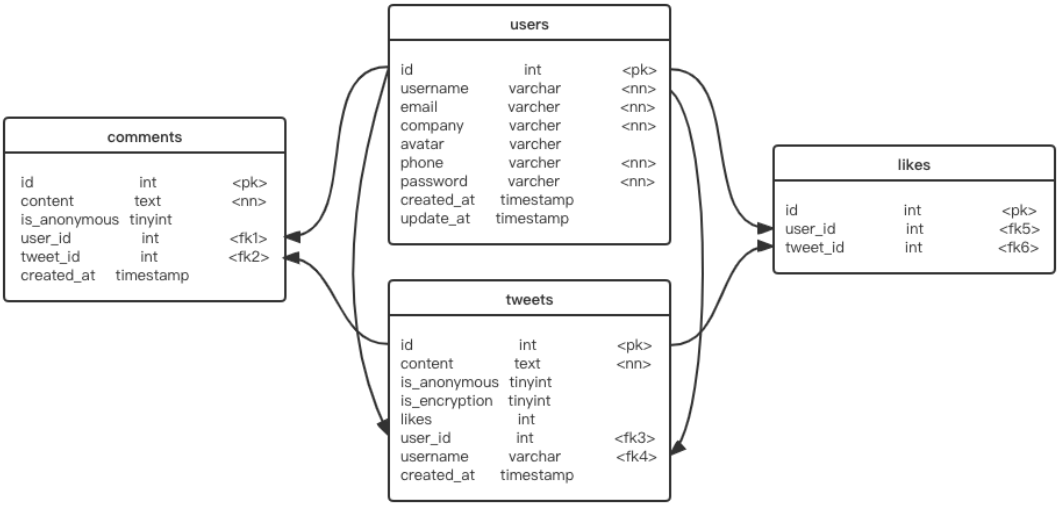


Figure 5: Database Design & Foreign Key.

4.3 Data Layer

The data layer is responsible for database access and can read database files to access data located in persistent containers. In the data layer, the system receives data from the browser, processes it before passing it to the database. Data processing includes read/write database, data cache, data screening, data correlation, exception handling and custom function.

1. **Read/Write database:** Use Node.js to manipulate the database. The "users" table contains an API for adding, updating and searching data. Users can create and modify their accounts and personal information. The system can query user information based on user input to find an account to complete login. The "tweets" table contains an API for adding and searching, and users can publish posts or search posts according to keywords. The "comments" table includes the add API, which is called when the user comments. The "likes" table also only sets up the add API, which is called when the user thumbs up tweet.
2. **Data cache:** After a user logs in, the system caches the current user's data until the user logs out. A system contains multiple sub-applications, each of which requires the same authentication. Therefore, to avoid frequent database

queries and improve efficiency, use sessions or tokens for identity authentication.

3. **Data Correlation:** According to the command of the database, design foreign keys. Associate users' comments and likes with the current posts' ID.
4. **Custom functions:** Including sensitive information detection and NER.
5. **Exception handling:** If the input data is incorrect, the system throws an exception and prompts the user.

4.4 Function Layer

All the specific data table designs are displayed in the appendix. The overall design and foreign key relationship of the database are shown below. The functional layer is divided into seven parts: Basic Function, Send Post, Information Detection, Read Post, Decrypt Post, Comment Post, and Search Post.

4.4.1 Basic Function

Users can register and log in and update basic personal information. The system would encrypt the user's password and transfer it to the database.

4.4.2 Send Post

In the post-editing box, users can add emojis when posting and can choose to send posts anonymously or encrypt them. Anonymous sending is when a user posts, the username is not displayed, the avatar is overwritten with the default picture. Encrypted sending is not only anonymous but also the content of the post is encrypted.

4.4.3 Information Detection

If users are unsure about their content, they can use the system's sensitive information detection function. Before sending a post, users can click the "check" button to verify whether the post content contains sensitive information. The sensitive information detection system consists of two parts. The first part is sensitive information detection of the content of posts, and the second part is entity detection using the NER tool.

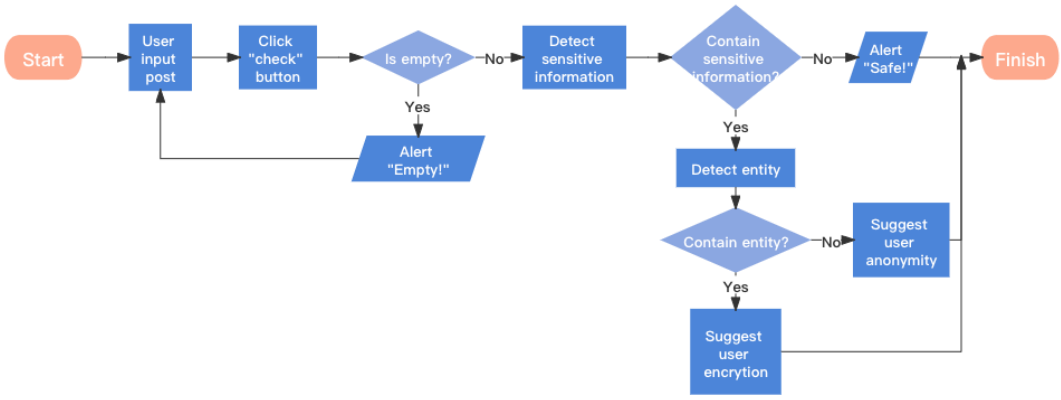


Figure 6: Sensitive information detection mechanism.

Sensitive information detection is to identify the text content of a post to determine whether there is inappropriate content. The scope of the test includes:

1. Negative workplace news
2. Complain about your boss or workmates
3. Talk openly about salary
4. Reveal your job-hop plans
5. Insulting language

The scope of the NER tool includes name, salary, location, and company name.

Suppose the system detects inappropriate statements but does not detect entity information, which means that based on the content of this post alone, the user will not be exposed to any personal or surrounding information. In theory, no one can identify users of sensitive posts based on their content. As a result, users are notified of the leak of sensitive information and advised to post it anonymously. As long as they remain anonymous, other users cannot get any information about the author of a sensitive post.

Another situation is, the system detects sensitive information and entity information. In other words, personal information or surrounding information might be leaked through this post. Other users may identify the user based on the sensitive content and the entity information the user discloses. Therefore, the system would prompt they may make sensitive information leakage and advise users to encrypt it for publication. All of the encrypted messages can only be seen by employees of other companies.

The above is a reasonably sensitive information detection logic obtained after discussion with the tutor. However, the most critical problem is implementing it, which will be covered in Section5.2.3. This includes training the model and applying machine learning to JavaScript projects, which is one of the most significant difficulties in implementing SafeTweet.

4.4.4 Read Post

Users can read all public posts, anonymous posts and can read encrypted posts from other companies' employees. Users can thumb up posts they like.

4.4.5 Decrypt Post

Users can read all public posts, anonymous posts and can read encrypted posts from other companies' employees. Users can thumb up posts they like.

Users can read and decrypt encrypted posts sent by employees of other companies and obtain the original post content on the decryption interface. The system should design a scheme to prevent the spread of encrypted content to avoid the spread of encrypted information as much as possible. In addition to the simple prohibition of copying, the author came up with the idea of adding watermarks to the interface after a simple survey of people of all ages. 90% of people said they wouldn't share screenshots with their name and email watermarked in this survey. So, the scheme for avoiding spread is:

1. Disable replication
2. Disable right-click to open the menu bar of the browser
3. The name and email watermark of the user who read the post are tiled on the decryption interface.

4.4.6 Comment Post

Users can comment on any readable post, add emoticons, and remain anonymous.

4.4.7 Search Post

Users can search for posts based on keywords.

4.5 Display Layer and Front-end UI

All the specific data table designs are displayed in the appendix. The overall design and foreign key relationship of the database are shown below. The site prototype is displayed below.

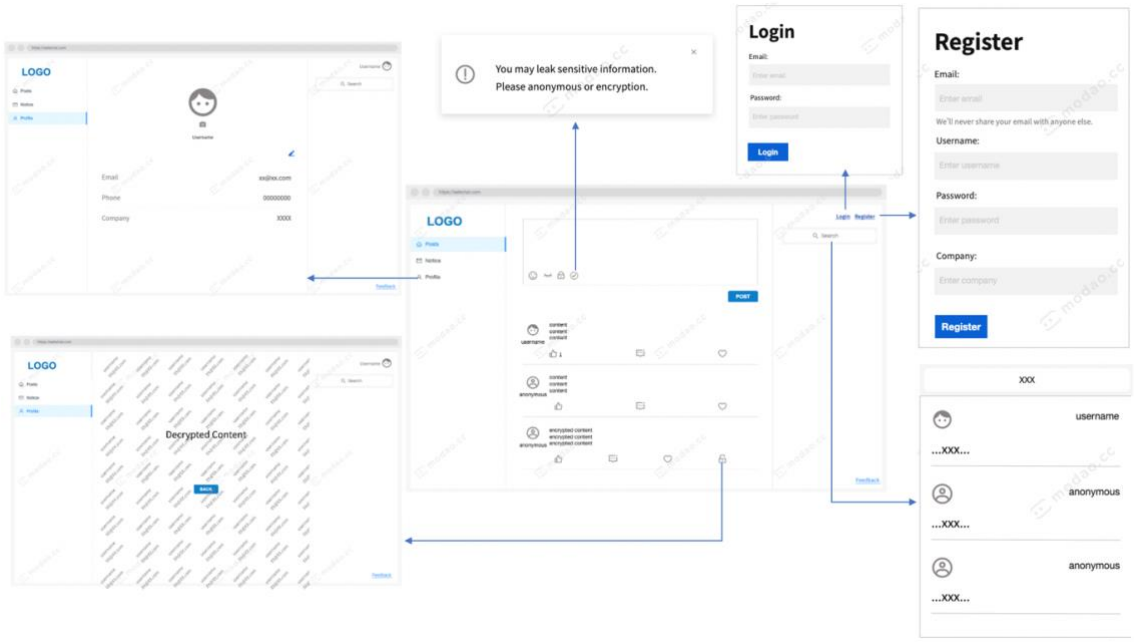


Figure 7: Main pages prototype.

The main page of the site is a three-column distributed display. The left column is the navigation bar with three modules. The first part is the home page, the second is the system notification, and the third is the personal information page. The middle column is the area for posting and interacting with other posts. The right column contains the login, register, logout, and search sections.

Chapter 5 Implementation

This chapter mainly introduces essential parts of the building and the challenges encountered and solutions. In order to explain the implementation process more clearly, this chapter presents the performance in the same order as the sequence of system development. Firstly, the author introduces the construction of the database (Section 5.1), then the development of the function (Section 5.2). The function section includes four challenging modules, Basic function, Send Post, Information Detection and Read Post module.

5.1 Database

Create a local MySQL database named "Safetweets". The tables are not created at the terminal but via knex.js [20], an SQL constructor based on Node.js. To make the system more portable, knexs.js is used instead of creating tables directly on the terminal. Knex.js uses code to create data tables, which can be quickly created by running code when the entire project code is copied or moved to another computer or pulled from GitHub by someone else.

5.2 Function Layer

The vue realises the front-end, and the back-end is developed by Node.js. Introduce the modules following the sequence of function requirements: Basic Function, Send Post, Information Detection, Read Post, Decrypt Post, Comment Post and Search Post.

5.2.1 Basic Function

5.2.1.1 Encryption Password

The Basic Function part, including user login and register, involves the security of data transmission. Specifically, the system should encrypt users' passwords during login and registration before transmitting them to the database. It is an approach to prevent database leakage or SQL injection attacks from exposing users' passwords. The hash encryption algorithm is a good choice because it is irreversible; the developer cannot deduce the original text from the ciphertext after the encryption is transmitted to the server. Standard hash encryption methods include MD5 encryption [21]. However, the biggest problem of this algorithm is that there will be collisions; that is to say, different texts can get the same password [22]. If the original text is M1, you only need another password with the same hash value to log in.

$$\mathrm{MD5(M1)} = \mathrm{MD5(M2)} = \mathrm{MD5(M3)}$$

The solution to this problem is to add salt. Adding additional information to the text before encrypting it is called salting it. The mixed information is not stored in the database, so attackers cannot log in even if they find another text with the same hash. Therefore, based on Node.js, bcrypt using the one-way hash algorithm is selected as the encryption method [23]. The encryption text connects bcrypt version number, salt and hash. The stitching method is shown in the figure.

Figure 8: Bcrypt hash encryption.

The characteristics of the bcrypt are that the hash value is different each time, and the calculation is very slow. Therefore, when an attacker wants to use rainbow tables for hash collisions, the time cost of attacking bcrypt is much higher than that of attacking MD5. Although bcrypt can compromise system performance to a certain extent, login, do not occur all the time and therefore can be within the acceptable range of loss.

5.2.1.2 Cross-domain Authentication Scheme

Currently, commonly used cross-domain schemes are session-based and token-based authentication. The most common token-based is the JSON Web Token (JWT). The verification process of these two methods is as follows:

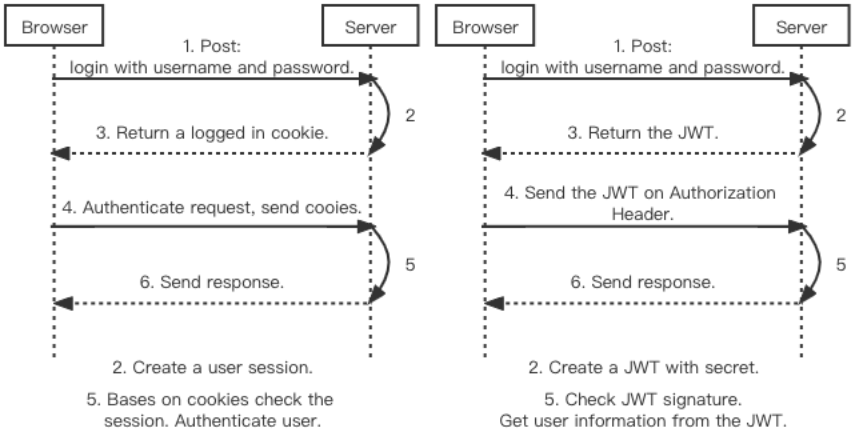


Figure 9: Session Authentication. **Figure 10:** Token Authentication.

In the session authentication method, the user information needs to be stored on the server for the first login and subsequent requests, which increases the overhead server [24]. JWT stores the user state on the client-side. As long as the user's information is validated, all subsequent requests from the user can be authenticated by JWT to access the server-side API, which significantly reduces the server-side memory stress [25]. Therefore, JWT is chosen as the cross-domain authentication scheme.

5.2.1.3 Request Method

The front-end and back-end send requests mainly through JavaScript's native jQuery ajax and the third-party library Axios. Ajax is already a mature and standardised request method for communicating with the server without refreshing the entire page. However, its disadvantages are also apparent; the Web-side is vulnerable to attacks, such as SQL injection. Axios is a more powerful encapsulation of Ajax, running in both a browser and Node.js. Security is also

enhanced, and clients support protection against CSRF. Therefore, Axios is chosen as the method of cross-domain request for this project.

5.2.2 Send Post

Due to not finding a suitable open-source rich text editor, the author developed a simple text box. The text box contains four buttons for emojis, anonymity, encryption and detection. Since jQuery was not introduced in vue, the text editing box handles the bi-directional binding of content via the *v-model*.

Figure 11: Editing box.

Emoji button: Vue-emoji-picker is used after comparing multiple emoji packages, which is a highly customisable plug-in. Use *v-show* to control its display and hiding. Reposition the cursor after each insert.

Anonymous button: Verify that the user is anonymous by using *v-if* in vue. Default to non-anonymous and set the initial value of "privacy" to False in the data source. If the user chooses anonymity, *v-if* changes the "privacy" to True. The system will hide the user's name and profile picture.

Encryption button: *v-if* is also used to determine, and when the users choose encryption, it will automatically anonymise users. The encrypt function does not in order to prohibit people from reading the plaintext, it does not need to carry out very complex encryption and decryption operations. Therefore, reversible encryption or encoding methods are required. Based on this requirement, the encryption part adopts Base64 encoding, a fast-transcoding method for representing binary data based on 64 printable characters [26].

Check button: Information detection.

5.2.3 Information Detection

This section detects sensitive information against text analysis; hence, the author considers supervised machine learning. The training classifier by training users' tweets on Twitter to predict whether a user's posts contain sensitive information. The overall machine learning process is:

Figure 12: Process of text analysis.

1. Consider Keywords about Workplace

Since this project is on account of the workplace social network, it mainly considers the sensitive information in the workplace. The range of data crawled was 10% of general tweets and 90% of workplace tweets. While regular tweets can be crawled randomly, workplace tweets need to be narrowed down by keywords. Through consulting tutor, searching literatures and my own understanding, I chose keywords: "job", "work", "overtime", "boss", "employer", "colleague", "workmate", "salary", "wage", "income", "burnout", "equality", "get fired" and "get the sack".

2. Crawler Data

Crawler data uses Twitter's official API V2 and Postman, a platform for building and using APIs. Firstly, sign up for a Twitter developer account for authorisation. Then choose a method between “search_recent_tweets” and “search_all_tweets” to search data. The former searches tweets within the past seven days, while the latter has a wide range. To gain more typical information, I chose “search_all_tweets for this project. Set the start time and end time of the search range between January 1, 2020, and October 1, 2021, to crawl 8000 pieces of data from Postman and save the obtained .json file as a .xlsx file for easy annotation.

3. Annotation data

Since a large number of tweets are non-sensitive, most of the non-sensitive data are filtered out after several rounds of screening to balance sensitive and non-sensitive data. After manual annotation, total obtain 800 valid data. Then randomly select 650 valuable data and store them in excel. It contains 280 sensitive and 370 non-sensitive data as the final data set.

4. Data Preprocessing

Data preprocessing includes determining the English string, removing punctuation, word stem restoration, and filtering stop words. This step relies heavily on the nltk library, a Python Natural Language Toolkit [27]. Nltk can meet almost all data preprocessing requirements. Firstly, regular filter all pure English strings and extract the word stem. Then, all the punctuation would be removed. Besides, all the URLs, starting with “http”, are also deleted. As for the stop words, combine nltk's stop words table with the stop words I set to remove all the stop words in the valid data.

5. Feature Extraction

Carry out vectorisation and feature extraction for all data by the sklearn library. Sklearn is a powerful Python machine learning library covering everything from data vectorisation to training models. For feature extraction of data, need to calculate term frequency (TF) and inverse document frequency (IDF). Firstly, use CountVectorizer() method to convert sensitive and non-sensitive data into vector form respectively, generate sparse matrix, and form a dictionary. This step is to get the term frequency in the data. Then calculate the inverse document frequency by using the TfidfTransformer() method.

6. Train Model

Usual text classification models include SVM, K-NearestNeighbour (K-NN), Naïve Bayes, Decision Tree, Adaboost and Random Forest.

SVM: SVM is a binary classification model. Its basic model is a linear classifier with the most considerable interval in the feature space. SVM contains different

kernel functions to solve the problem of linear inseparability in real data [28]. I use linear kernel and poly kernel to train the SVM model.

K-NN: In the feature space, if most of the K nearest samples near a sample belong to a specific category, then this one also belongs to that category. After adjusting the number of neighbours, it will have the maximum accuracy when the number of neighbours is 13.

Neighbours	5	7	9	11	13	15
Accuracy	74.62%	73.08%	73.08%	73.84%	76.15%	72.31%

Figure 13: Accuracy of different Neighbours.

Naïve Bayes: A simple probabilistic classifier based on Bayes' theorem. The Naïve Bayes dependence requires Laplace smoothing to deal with possible zero-probability problems [29], so set a number for the Laplace smoothing parameter. Since the sensitive and non-sensitive information in the data set is not balanced, the prior probability needs to be considered by the Bayesian model.

Decision tree: A method to approximate the value of the discrete function. Due to a large amount of data, and to prevent excessive fitting, set the tree depth to 10. Placing the minimum number of samples required to 4 would obtain the highest accuracy [30].

Adaboost: An iterative algorithm whose core idea is to train different classifiers for the same training set and then assemble these weak classifiers to form a more robust final classifier. Set the number of iterations of the parameter to 100. When the learning rate is 80%, it achieves the highest accuracy [31].

Random forest: A classifier that uses multiple trees to train and predict samples. The appearance of a random forest can solve the weak generalisation ability of the decision tree. The difference between this algorithm and Adaboost is that the pieces of random forest are randomly selected, and the training samples of almost every tree are different [32].

Compare all models against four criteria: accuracy, precision, recall and classification duration. The final results of each classifier are:

Model	**SVM (LINEAR)**	**SVM (poly)**	**KNN_N13**	**Naïve Bayes**	**Decide Tree**	**Adaboost**	**Random Forest**
Accuracy	81.54%	63.08%	76.15%	82.31%	73.08%	70.77%	70.00%
Precision	66.04%	9.43%	62.26%	66.04%	47.17%	75.47%	35.85%
Recall	85.37%	100.00%	75.00%	87.50%	78.13%	61.54%	79.17%
Time	0.04s	0.04s	0.02s	0.00s	0.00s	0.03s	0.02s

Figure 14: Performance of different models.

The results in Figure 14 show that Naïve Bayes performs best in Accuracy. In the Figure 15, the ROC curve indicates Naïve Bayes performs best. Therefore, adjust the parameters of Naïve Bayes, and the model achieved the highest accuracy when the Laplacian smooth adjustment is set to 1.1.

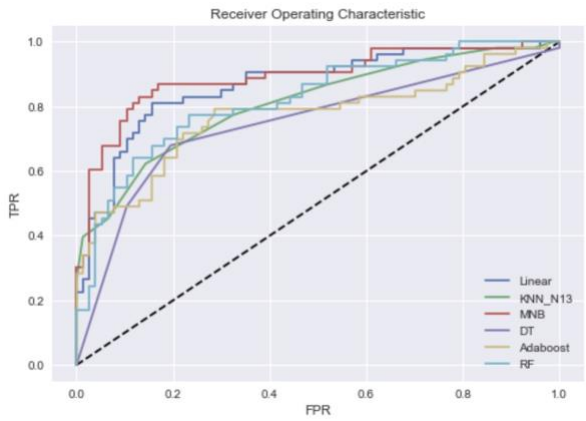


Figure 14: ROC curve of different models.

Model	Accuracy	Precision	Recall	Time
Naïve Bayes	83.85%	67.93%	90.00%	0.00s

Figure 15: Performance of Naïve Bayes.

7. Save and Call Model

The system should not re-train the model each time users detect posts to save detection time. The learned model should be saved and directly called when used. Joblib is a package that can fulfil this requirement. After using Joblib to save the model to the clients, the Abspath can directly obtain the model's path in different devices. One difficulty with model calls is how to run Python code in JavaScript. The author found a "python-shell" package that allows running Python code in a JavaScript project with appropriate parameter settings.

Therefore, the authors find that it is possible to introduce machine learning into JavaScript projects. The first step is to introduce the third-party libraries needed for machine learning into the project. Joblib and Abspath are used to store and invoke models in the client. Finally, use "python-shell" to run the Python model called on a JavaScript project.

5.2.4 Read Post

To prevent users from seeing encrypted messages from other users of the same company. Whenever a user acquires or updates a job list, the system filters all encrypted information belonging to the company's employees except for the user. When the system calls the API for obtaining or updating tweets, it will get the ID and company information of the current user and pass them to the back end. The SQL statement logic executed in the back section is:

1. Output all tweets with encryption value 0 from the tweets table (marked 1 for encrypted tweets)
2. Output from the tweets table all tweets with encryption value 1 and company value not equal to the company value passed in from the front end
3. Output all tweets from the tweets table with encryption value 1, company value equal to the company value passed in and ID equal to the user ID given in.

The first step is to get all regular and anonymous tweets, the second step is to get encrypted tweets from other companies, and the third step is to get encrypted tweets from the user.

Chapter 6 Test and Evaluation

This section contains functional testing (Section 6.1) and user evaluation (Section 6.2). The user evaluation is divided into two parts, the evaluator according to the task table operating system and the questionnaire after the task. The author makes an analysis based on the evaluation results (Section 6.3). Finally, it introduces the future work about SafeTweet (Section 6.4).

6.1 Testing

The software testing uses Selenium Python to implement basic automated testing. Also need to use WebDriver, a third-party library, to implement web test automation. Selenium automatically manipulates browsers, simulating interactions with browsers and supports most major browsers. Selenium commands fall into three categories: action, accessors, and assertion.

1. Action: Simulate user interaction with a Web application.
2. Accessors: Check the application's state and store the results in variables.
3. Assertion: It is a Boolean expression. I use Python's native Assert in the tests. If any assertion fails, the script execution stops. Click on links and select options to work. If an action fails or an error occurs, the current test will stop execution.

When using Selenium for software testing, need to use assertions reasonably. When all assertions pass, the test passes. Besides, sleep time is another concern. Duo to many operations needs to rely on the results or content of the previous step, such as text encryption and information detection. In particular, it takes a long time to detect sensitive information, so it is needed to set a mandatory waiting time.

The main test content of the system is whether the front-end page is the correct jump, whether the back-end calls the API correctly, whether the operation of the database is successful. The test module is classified into Basic Function, Send Post, Information Detection, Read Post, Decrypt Post, Comment Post and Search Post, a unit test. This is the number of test cases per module and the results.

Module	Test Cases	Assertions	Pass	Fail
Basic Function	3	4	4	0
Send Post	3	6	6	0
Information Detection	7	8	8	0
Read Post	1	2	2	0
Decrypt Post	1	2	2	0
Comment Post	2	5	5	0
Search Post	1	2	2	0

Figure 16: Test result.

6.2 User Evaluation

Testing only checks the system if there are undetected problems, but specific usability needs to be evaluated by real users. The primary determinant of user activity in social networking markets is product availability. Therefore, when a new product is born, user evaluation is needed to get feedback from different users on their experiences and feelings. However, user experience is a purely subjective psychological feeling, with many uncertainties and individual differences. It is not easy to accurately evaluate user experience. In order to accurately assess the user experience, all evaluation criteria should be quantifiable, measurable, observable and reproducible [33].

6.2.1 Evaluator

There are five evaluators, all of whom are fans of social networking and have accounts in all major social networks. All evaluators are students at the University of Glasgow. Three of them majoring in computer science, the other two are business students. Two students had previous work experience. The authors chose an evaluator with experience testing products for the first round of testing. This is to get initial feedback to adjust the evaluation plan so that you can get better feedback from other evaluators.

6.2.2 Evaluation Processes

Divide the user evaluation into two parts. The first part is to complete the task according to the task list. The task list includes all the functions of the system, which need the evaluator to accomplish. According to the completion of the task list, we can evaluate the usability and learnability of the system. The task sheet is available in appendix C. The second part is the questionnaire at the end of the task. The questionnaire is based on Brooke's testing table [34], and the author made modifications according to the characteristics of SafeTweet. The contents of Brooke's ten scoring questions were partially modified in the word order, and the question sentences were all changed into positive ones. This can better obtain the average score for analysis. At the same time, the contents of some questions in the questionnaire were modified in combination with SafeTweet. In addition, the authors add eight questions based on usability, learnability, memorability and efficiency [35], which are used to dissect better how evaluators feel about each module in SafeTweet. It is for analysing user experience and satisfaction. The questionnaire is in appendix D.

6.2.2.1 Assessment steps

1. The assessment time for all evaluators is 9:00-12:00 in a quiet living room. Each assessment is separate.
2. The author introduces the evaluator to SafeTweet's background, main interface, and features through a pre-prepared system overview slide.
3. The author introduces the task list to the evaluator, who reads the task list. If you have any questions, please feel free to ask the author to answer them.
4. Enable screen recording to record the execution duration of each task.
5. The evaluator completed the task one by one, and each task completed was recorded as task success. If the evaluator abandons an assignment, then this assignment fails.
6. After the task, ask the evaluators to fill in the questionnaire.

7. Collect valid task lists and questionnaires, and record the execution duration of each task according to screen recording. A complete task list is that 80% of the tasks have been completed, which can be valid data for analysis.

6.3 Result Summary

The results are presented and analysed in two parts. The first section is how long it took each evaluator to complete each task and the average time. The second part is the questionnaire filled in by the evaluators after the evaluation. The questionnaire analysis was presented using the average score for each item and an overview of the subjective questions.

ID	Duration of Evaluators Execute Tasks(s)					Average Duration
	E1	E2	E3	E4	E5	
A. Basic Function						
1	37	44	40	35	42	39.6
2	15	10	11	11	9	11.2
3	53	21	20	19	29	28.4
4	40	29	30	18	21	27.6
B. Send Post						
1	8	12	17	19	14	14.0
2	10	15	11	10	13	11.8
3	8	11	10	22	17	13.6
C. Information Detection						
1	37	48	39	46	51	44.2
2	21	33	27	31	28	28.0
D. Read Post						
1	1	2	1	1	1	1.2
E. Decrypt Post						
1	6	6	5	13	5	7.0
F. Comment Post						
1	7	5	11	5	4	6.4
2	4	5	12	6	4	6.2
G. Search Post						
1	3	3	4	5	4	3.8

Figure 17: Results of evaluation task.

According to Figure 17, each evaluator's execution time and average duration of each task can be obtained. Due to the extensive experience of all the evaluators in using social platforms, all the tasks were successfully performed. Next, the single task with a large gap between the average time and the task with a long average time are discussed.

1. A3: Upload a profile picture. In the A3 task, the first evaluator took longer. The system has completed the avatar upload, but there is no indication of success, so the evaluator has been waiting for the result.

2. E1: Decrypt a tweet. In the D1 task, the fourth evaluator had no clear perception of the encrypted text and could not find the encrypted tweet accurately.
3. C1: Enter a sensitive message for detection. B4 has the most extended average duration of all tasks. Because evaluators generally do not have a clear definition of sensitive information in the workplace, they often have to think for a long time before coming up with a tweet that contains sensitive information about the workplace. The slow detection speed also has an impact on the execution time.

The results of the questionnaire assessment section are in Figure 18.

ID	Question	Average Score
1	I think that I would like to use this system frequently	3.8
2	I think the complex of the system is reasonable	4.4
3	I think the system is easy to use	4.4
4	I think that I can use this system without the support of a technical person	5.0
5	I think the various functions in this system are well integrated	4.2
6	I think the system is consistency	4.2
7	I think that most people would learn to use this system very quickly	4.4
8	I feel very confident using the system	3.8
9	I do not need to learn a lot of things before I could get going with this system	4.2

Figure 18: Partial results of the questionnaire.

The two low-mark questions in part one are question 1 and question 8. The evaluators were sceptical about whether the system could be used on a large scale and become a mainstream social network, mainly because of their subjective responses. The second part of the questionnaire results are as follows; The module elects each section with the most votes.

Performance	Usability	Learnability	Memorization	Efficiency
Worst	Information Detection	Information Detection	Search	Information Detection
Best	Decryption	Search	Information Detection & Send Post	Read

Based on the above conclusions, all evaluators agreed that the system was workable, easy to learn and easy to use. However, the main question is whether people need such a social network. SafeTweet's original design vision of creating a secure workplace social network where users can safely share workplace updates or secrets will only be realised when the number of users reaches millions. The modules that the evaluators thought performed well were:

1. Anonymity works great when posting.
2. The encryption function is unique, an innovation not found in other social networks.
3. High accuracy of the detection function.

4. A series of measures to prevent private posts transmission is very effective.
5. The anonymous module in the comments section is also handy.

Modules that are not performing well are:
1. The detection time is long.
2. Some icons are hard to understand.
3. Encryption is not very useful.
4. Too few ways to interact with other users.
5. Few sensitive information cannot be detected.

6.4 Future Work

In future work, the first direction of continuous efforts should be to improve the efficiency of sensitive information detection. The model detection time is due to the long-running and calling time of the model. This needs to be addressed by optimising the algorithm and adopting better model invocation methods. The optimisation algorithm can improve the speed by modifying the practice and process of data pretreatment. The model call method is the current faster call method that the author tried. A more suitable model call package may be needed to solve this problem, or the project will be developed with The Python framework Django in the future. The second thing to tackle is more elegant encryption. SafeTweet currently uses Base64 for plain text transcoding in a sloppy way. If asymmetric encryption can be adopted to transfer the key securely, the existence and availability of encryption will be significantly improved. In addition, how-to guide users to use SafeTweet with their real names is also a problem that needs to be considered. This may be helped by adding recruitment and job-hunting functions to SafeTweet. Finally, I will continue to complete more essential functions of social network platforms, such as following other users, uploading pictures and videos, etc.

Chapter 7 Conclusion

SafeTweet is a web-based, real-name social network for the workplace. SafeTweet differs from other major social networks in that it allows people to anonymously post sensitive information about their company and shield colleagues or bosses from anonymous, encrypted posts. This project fulfilled most of the requirements analysis demand as far as possible. All the needs were collected from people keen on social networks, and some of the interviewees had work experience.

At the beginning of the design, the basic system architecture was established, and the prototype diagram of the main interface was designed. Specific database tables, primary keys, foreign keys, and indexes for each table are designed according to requirements. Each particular function was designed by analysing the criteria sorted in the MoSCoW method and combining them with reality. The development of SafeTweet started with deciding tools. The front end uses the Vue framework, the back end uses Node.js, knex as middleware, and connects to MySQL database. The most critical sensitive information detection function is realised by machine learning. After comparing several commonly used text analysis models, Naïve Bayes with the best performance is selected as the sensitive information detection model. The machine learning model has been successfully deployed to JavaScript projects through the proper use of various methods. Several asynchronous operations were used throughout the project to improve database execution efficiency. Most of the functional design and page construction has been completed.

Testing and evaluation are divided into two parts. The first part is about running the software's core functions through Selenium automated tests, and the second part is about user evaluation. Five people active in social networks were asked to evaluate the system. Each evaluator interacts with the system based on a task list. The analysis was based on the average time spent on each task and the feedback from the evaluators. The results showed that the reviewers found SafeTweet valuable and easy to use, with powerful built-in sensitive information detection capabilities and attractive features such as anonymity and encryption. However, the main problems are the long detection time and the impracticality of the encryption function. Then, the author thinks about the current issues of SafeTweet in the future. This includes model optimisation at the technical level and interfaces beautification at the usability level. In the final version, the author modified interfaces according to user experience.

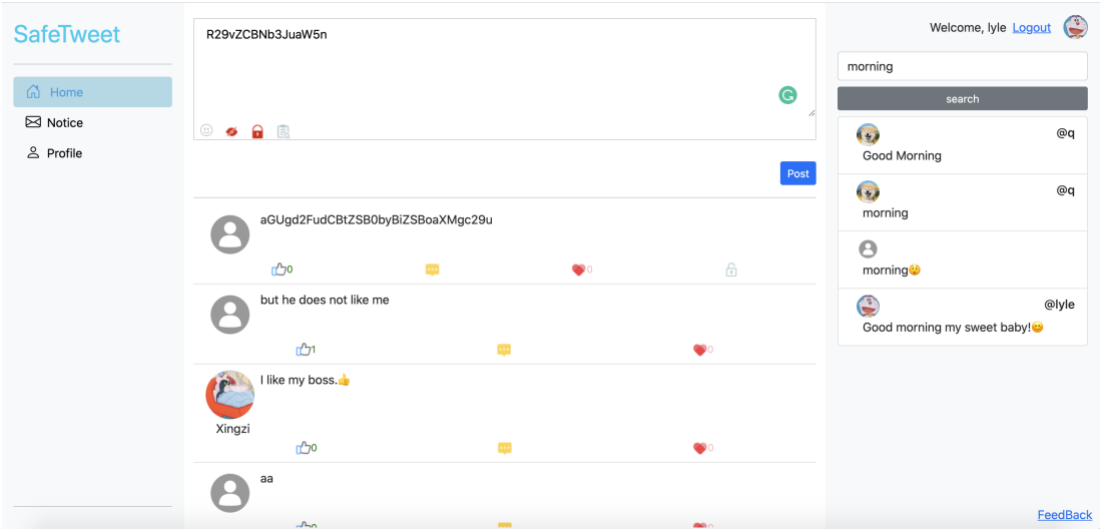
SafeTweet is a groundbreaking new social network that meets people's need to communicate about their careers safely and openly. It even might improve people’s chances of finding a suitable job. After much polishing, SafeTweet may become a robust and mature social network in the future.

Appendix A Requirement List

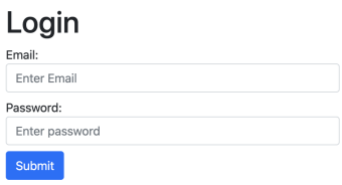
ID	Module	Description	MoS CoW	Implemented
1	Basic Function	User can register	M	Y
2		User can login	M	Y
3		System should encrypt users' passwords	S	Y
4		Users can upload profile pictures	S	Y
5		Users can modify personal information	M	Y
6	Send Post	Users can send posts directly	M	Y
7		Users can insert emoticons in posts	C	Y
8		Users can send posts anonymously	M	Y
9		Users can send encrypted posts	M	Y
10	Information Detection	User can detect sensitive information	M	Y
11		System must response the detection result	M	Y
12		System must suggest the sending method	M	Y
13	Read Post	Users can view the avatar and name of the publisher	M	Y
14		Users can read all direct and anonymous posts	M	Y
15		Users can only view encrypted posts from employees of other companies	M	Y
16		User can like posts	S	Y
17	Decrypt Post	Users can decrypt posts from employees of other companies	M	Y
18		System could place the watermark of the user's name and email on the decryption interface	C	Y
19		System must forbid users to copy content on the decryption page	S	Y
20	Comment Post	Users can view post comments	M	Y
21		Users can comment on post	S	Y
22		Users can insert emojis into comments	C	Y
23		Users can make comment anonymously	S	Y
24		Users can encrypt their comments	W	N
25	Search Post	User can find posts by entering keywords	S	Y
26		System should display the content and publisher name for non-anonymous posts	S	Y
27		System must display only the content of anonymous posts	M	Y

Appendix B Screenshots

Main Page

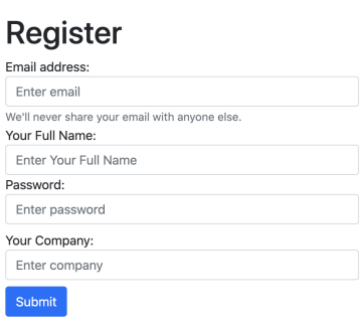


Basic Function

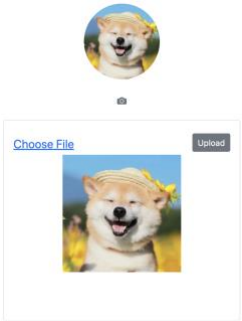


Login

Update Information

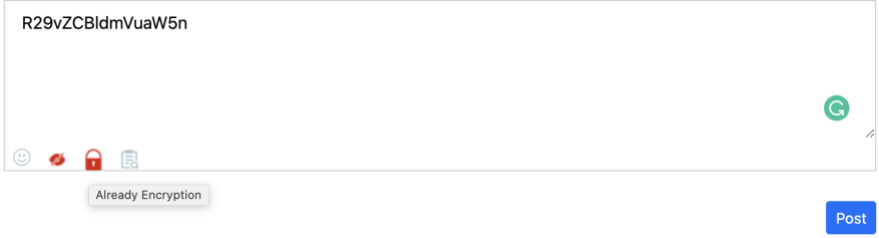


Register



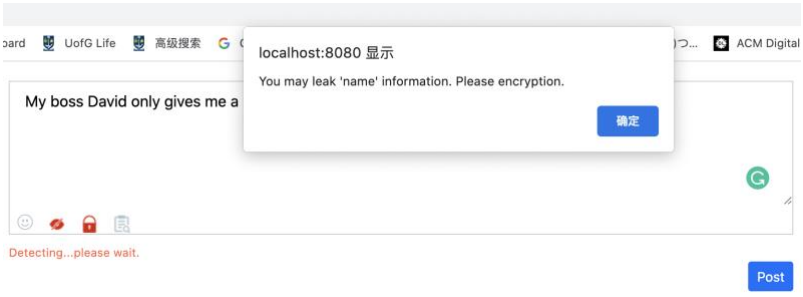
Upload Avatar

Encrypt Posts



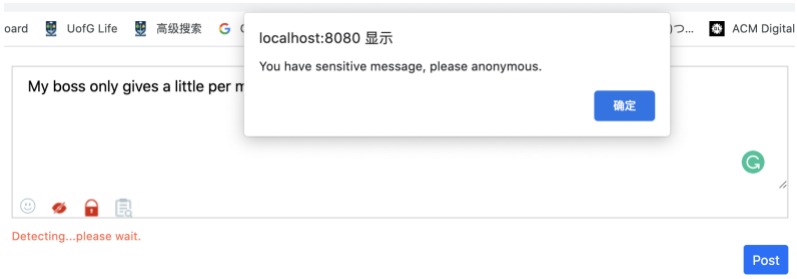
Transcoding “Good Morning” through Base64 to the above garble.

Information Detection



Detect sentence: My boss David only gives me a little per month.

The return result is: You may leak “name” information. Please encryption.



Detect sentence: My boss only gives me a little per month.

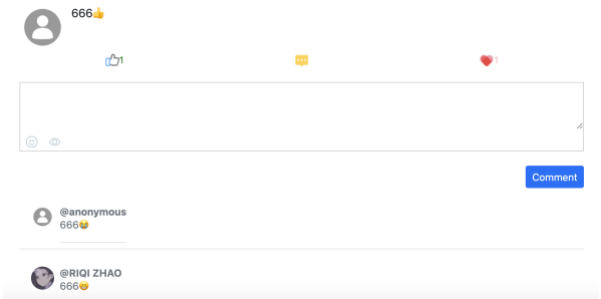
The return result is: You may have a sensitive message, please anonymous.

Decrypt Posts

In this interface:

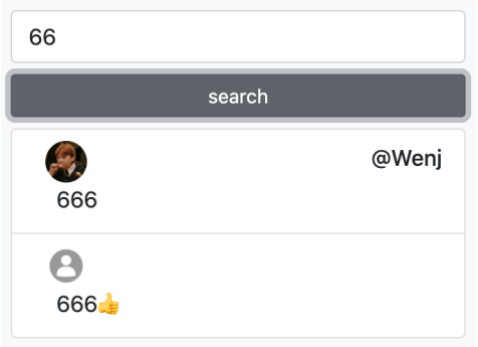
1. Prohibit Copy.
2. Prohibit right-click to invoke the Google menu.

Comment Posts



Users can be anonymous when commenting.

Search Posts



Users can be search posts according keywords.

Appendix C Task List

ID	Test Content	Pass	Duration
A. Basic Function			
1	Register		
2	Log		
3	Upload an avatar		
4	Change Phone Number		
B. Send Post			
1	Send a post		
2	Send a post anonymously		
3	Send an encrypted post		
C. Information Detection			
4	Input sensitive information in text box and check sensitive information		
5	Input non-sensitive information in text box and check sensitive information		
D. Read Post			
1	Thumb up a post		
E. Decrypt Post			
1	Decrypt a post		
F. Comment Post			
1	Comment a post with emojis		
2	Comment a post anonymously		
G. Search Post			
1	Search posts by entering keyword		

Appendix D Questionnaire

ID	Question	Score 1 - Strongly disagree 5 - Strongly Agree
1	I think that I would like to use this system frequently	
2	I think the complex of the system is reasonable	
3	I think the system is easy to use	
4	I think that I can use this system without the support of a technical person	
5	I think the various functions in this system are well integrated	
6	I think the system is consistency	
7	I think that most people would learn to use this system very quickly	
8	I feel very confident using the system	
9	I do not need to learn a lot of things before I could get going with this system	
10	I feel that the _____ modules are not performing well in terms of usability, but _____ performing well. a. Basic function b. Send Post c. Information Detection d. Read Post e. Decryption f. Comment g. Search Post Reason:	
11	I feel that the _____ modules are not performing well in terms of learnability, but _____ performing well. a. Basic function b. Send Post c. Information Detection d. Read Post e. Decryption f. Comment g. Search Post Reason:	
12	I feel that the _____ modules are not performing well in terms of memorization, but _____ performing well. a. Basic function b. Send Post c. Information Detection d. Read Post e. Decryption f. Comment g. Search Post Reason:	
13	I feel that the _____ modules are not performing well in terms of efficiency, but _____ performing well. a. Basic function b. Send Post c. Information Detection d. Read Post e. Decryption f. Comment g. Search Post Reason:	

Bibliography

[1]. Sohail SS, Khan MM, Alam MA. An Analysis of Twitter Users From The Perspective of Their Behavior, Language, Region and Development Indices--A Study of 80 Million Tweets. arXiv preprint arXiv:210510245. 2021.

[2]. Faisal M, Alsumait A. Social network privacy and trust concerns. Proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services; Ho Chi Minh City, Vietnam: Association for Computing Machinery; 2011. p. 416–9.

[3]. Piao Y, Ye K, Cui X. Privacy Inference Attack Against Users in Online Social Networks: A Literature Review. IEEE Access. 2021;9:40417-31.

[4]. Zheng F, Ma L, editors. A Multi-layered Friend Recommendation System on Twitter. 2021 The 13th International Conference on Computer Modeling and Simulation; 2021.

[5]. Gezici A, Ozay O. An Intersectional Analysis of COVID-19 Unemployment. Journal of Economics, Race, and Policy. 2020;3(4):270-81.

[6]. Chen X, Liu Y, Zhang L, Kenthapadi K. How LinkedIn Economic Graph Bonds Information and Product: Applications in LinkedIn Salary. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining; London, United Kingdom: Association for Computing Machinery; 2018. p. 120–9.

[7]. Kenthapadi K, Ambler S, Zhang L, Agarwal D. Bringing Salary Transparency to the World: Computing Robust Compensation Insights via LinkedIn Salary. Proceedings of the 2017 ACM on Conference on Information and Knowledge Management; Singapore, Singapore: Association for Computing Machinery; 2017. p. 447–55.

[8]. Mao H, Shuai X, Kapadia A. Loose tweets: an analysis of privacy leaks on twitter. Proceedings of the 10th annual ACM workshop on Privacy in the electronic society; Chicago, Illinois, USA: Association for Computing Machinery; 2011. p. 1–12.

[9]. Islam AC, Walsh J, Greenstadt R. Privacy Detective: Detecting Private Information and Collective Privacy Behavior in a Large Social Network. Proceedings of the 13th Workshop on Privacy in the Electronic Society; Scottsdale, Arizona, USA: Association for Computing Machinery; 2014. p. 35–46.

[10]. Cappellari P, Chun SA, Perelman M. A Tool for Automatic Assessment and Awareness of Privacy Disclosure. Proceedings of the 18th Annual International Conference on Digital Government Research; Staten Island, NY, USA: Association for Computing Machinery; 2017. p. 586–7.

[11]. Neerbeky J, Assentz I, Dolog P, editors. TABOO: Detecting Unstructured Sensitive Information Using Recursive Neural Networks. 2017 IEEE 33rd International Conference on Data Engineering (ICDE); 2017 19-22 April 2017.

[12]. Canfora G, Sorbo AD, Emanuele E, Forootani S, Visaggio CA. A Nlp-based Solution to Prevent from Privacy Leaks in Social Network Posts. Proceedings of the 13th International Conference on Availability, Reliability and Security; Hamburg, Germany: Association for Computing Machinery; 2018. p. Article 36.

[13]. Delcev S, Draskovic D, editors. Modern JavaScript frameworks: A survey study. 2018 Zooming Innovation in Consumer Technologies Conference (ZINC); 2018: IEEE.

[14]. Wohlgethan E. SupportingWeb Development Decisions by Comparing Three Major JavaScript Frameworks: Angular, React and Vue. js: Hochschule für Angewandte Wissenschaften Hamburg; 2018.

[15]. Chitra LP, Satapathy R, editors. Performance comparison and evaluation of Node.js and traditional web server (IIS). 2017 International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies (ICAMMAET); 2017 16-18 Feb. 2017.

[16]. Tilkov S, Vinoski S. Node.js: Using JavaScript to Build High-Performance Network Programs. IEEE Internet Computing. 2010;14(6):80-3.

[17]. Malloy BA, Power JF. An empirical analysis of the transition from Python 2 to Python 3. Empirical Software Engineering. 2019;24(2):751-78.

[18]. Letkowski J. Doing database design with MySQL. Journal of Technology Research. 2015;6:1.

[19]. Waters K. Prioritization using moscow. Agile Planning. 2009;12:31.

[20]. Nandaa A. Beginning API Development with Node. js: Build highly scalable, developer-friendly APIs for the modern web with JavaScript and Node. js: Packt Publishing Ltd; 2018.

[21]. Ali AM, Farhan AK. A novel improvement with an effective expansion to enhance the MD5 hash function for verification of a secure e-document. IEEE Access. 2020;8:80290-304.

[22]. Tian Y, Zhang K, Wang P, Zhang Y, Yang J. Add" Salt" MD5 Algorithm's FPGA Implementation. Procedia computer science. 2018;131:255-60.

[23]. Patra R, Patra S. Cryptography: A Quantitative Analysis of the Effectiveness of Various Password Storage Techniques. Journal of Student Research. 2021;10(3).

[24]. Dammak M, Boudia ORM, Messous MA, Senouci SM, Gransart C, editors. Token-based lightweight authentication to secure IoT networks. 2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC); 2019: IEEE.

[25]. Ahmed S, Mahmood Q, editors. An authentication based scheme for applications using JSON web token. 2019 22nd International Multitopic Conference (INMIC); 2019: IEEE.

[26]. Muła W, Lemire D. Base64 encoding and decoding at almost the speed of a memory copy. Software: Practice and Experience. 2020;50(2):89-97.

[27]. Millstein F. Natural language processing with python: natural language processing using NLTK: Frank Millstein; 2020.

[28]. Tharwat A. Parameter investigation of support vector machine classifier with kernel functions. Knowledge and Information Systems. 2019;61(3):1269-302.

[29]. Kalcheva N, Nikolov N, editors. Laplace Naive Bayes classifier in the classification of text in machine learning. 2020 International Conference on Biomedical Innovations and Applications (BIA); 2020: IEEE.

[30]. Charbuty B, Abdulazeez A. Classification based on decision tree algorithm for machine learning. Journal of Applied Science and Technology Trends. 2021;2(01):20-8.

[31]. Wang F, Li Z, He F, Wang R, Yu W, Nie F. Feature learning viewpoint of adaboost and a new algorithm. IEEE Access. 2019;7:149890-9.

[32]. Speiser JL, Miller ME, Tooze J, Ip E. A comparison of random forest variable selection methods for classification prediction modeling. Expert systems with applications. 2019;134:93-101.

[33]. Díaz-Oreiro I, López G, Quesada L, Guerrero LA, editors. Standardized questionnaires for user experience evaluation: A systematic literature review. Multidisciplinary Digital Publishing Institute Proceedings; 2019.

[34]. Kusic S. „SUS-A quick and dirty usability scale ". Iron and Steel Technology. 2018;15:41-7.

[35]. Desak GFP, editor List of Most Usability Evaluation in Mobile Application: A Systematic Literature Review. 2020 International Conference on Information Management and Technology (ICIMTech); 2020: IEEE.