

<Title of project placed here>

<Name of author placed here>

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

<Date of submission placed here>

**Abstract**

<Abstract goes here…>

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.

<**Please note that you are under no obligation to sign this declaration, but doing so would help future students.>**

Name: Signature:

Acknowledgements

<Acknowledgements go here>

Contents

<Update the table of contents by right-clicking on it and selecting Update Field… and then select page numbers only.>

Chapter 1 Introduction <This is Heading 1> 1

1.1 A section <This is style Heading 2> 1

1.1.1 A subsection <This is style Heading 3> 1

Chapter 2 Survey 2

Chapter 3 Further Chapters 3

Chapter 4 Conclusion 4

Appendix A <Name of appendix> 1

Appendix B <Another appendix> 2

Bibliography 3

# Introduction

## 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. Concerning Twitter, it is the second-largest social network, has more than 1 billion registered users and 326 million active users [1]. It proves that people rely heavily on social networks. 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]. 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 [3]. So many ways can lead to leakage of improper comments. Employees who post sensitive information might face pressure from their companies if their inappropriate statements are discovered by companies. Some professionals use non-public social accounts to send posts. But if a post includes any name, location, or company can also be easily recognized 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% [4]. Not only because of the depressed market, information inequality 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. And 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 realize it until he starts working. There is also information inequality in terms of 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 is a system that gathers salary information from members to provide salary insights to job seekers [5]. In 2017, Kenthapadi proposed the LinkedIn compensation product, which helps people calculate their earning potential by collecting a large amount of data [6]. 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.

## Solution

SafeChat is a Web-based social networking platform that detects sensitive information in real-time. In SafeChat, authenticated users can post sensitive information without fear of being discovered by their bosses. Users are required to fill out basic information and the company they belong to when signing up for the social networking platform. Unlike other social networks, SafeChat allows people to choose to post anonymously, allowing them to hide their identities while posting. In addition, people can choose to encrypt posts through the Base64 algorithm. If the user encrypts the content, he is also anonymous automatically. All encrypted information cannot be seen by other employees of the same company. When other people read this post, they can click the decryption button next to it, then navigate into the decryption interface. The decryption page contains decrypted contents of tweets, which is forbidden to copy. 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. When users publish information, the system will automatically detect sensitive information. If there is sensitive information, NER detection will be performed on it, with the main detection objects being name, organization, location and money. If the post contains sensitive information, but the NER tool does not detect identity information, the system prompts the user to remain anonymous. If sensitive information and entity information are both distinguished, the system prompts the user to encrypt it.

# Related Work and Tools

## Related Work

This section mainly describes the work related to sensitive information detection.

In 2011, Mao proposed three types of tweets that could leak privacy and are worthy of attention [7]. Vacation tweets, drunk tweets and illness tweets. He used naive Bayes and the SVM classifier to classify sensitive information. The experiments indicate that the naive Bayes performed better than SVM. The accuracy of holiday tweets is 76% in naive 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 [8]. 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 naive Bayes had a better performance than in 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 [9]. He used five algorisms, including nearest neighbour, naive 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 RNN [10]. However, the author does not provide specific data for model accuracy. In 2018, Canfora used NLP to detect sensitive information in social networks [11]. The method is to judge sensitive information by analyzing 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, which is 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.

## 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 can 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 is to control the behaviour of the Web page [12]. JavaScript is one of three languages that Web developers must learn, along with 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 componentized 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 [13]. 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 suitable sever-side for real-time applications and multiple front-end technologies such as Vue and React [14]. It uses an event-driven, non-blocking I/O model, making it lightweight and efficient [15]. Currently, versions 12.x and 14.x are 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 in this project to train model, detect sensitive information and use NER tools. Python3 is a stable version that is not compatible with Python2.x [16].

**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 run-on different systems. It is the most widely used relational database management system [17]. 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.

# Requirements

## Requirement Gathering

Collect requirements through interviews at the beginning of the project. The purpose of the interview is to understand the behavioural characteristics and preferences of employees using social platforms. And their pain points in using social networks. 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 SafeChat 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 SafeChat.

The core users of this research object are two employees of Internet companies and an 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. During the interview, supplementary questions can be asked by adjusting the space flexibly. After the interview, collate all content for the requirements design of SafeChat and prioritize functions using Moscow. Interview questions are in the appendix.

Figure 1 is the classification of the Moscow method [18]. Figure 2 shows the number of MoSCoW for each functional module. The detailed requirement list is in the appendix.

|  |  |
| --- | --- |
| **Category** | **Criteria** |
| **M**ust | **Must** have requirement |
| **S**hould | **Should** have if at all possible |
| **C**ould | **Could** have but not Critical |
| **W**on'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 Functions | 3 | 2 | 0 | 0 | 5 |
| Send Posts | 7 | 0 | 1 | 0 | 8 |
| Read Posts | 3 | 1 | 0 | 0 | 4 |
| Decrypt Posts | 1 | 1 | 1 | 0 | 3 |
| Comment Posts | 1 | 2 | 1 | 1 | 5 |
| Search Posts | 1 | 2 | 0 | 0 | 3 |

Figure 2: MoSCoW Criteria.

# Design

According to the analysis of system requirements, design the overall structure of the system. The Web system is divided into six parts. Starting from the bottom are the run environment, database, data layer, function layer, display layer and front-end UI. The system structure design is as follow.

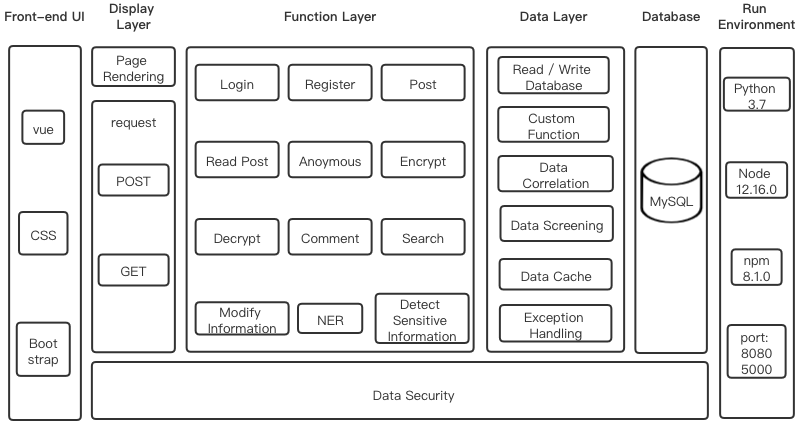
****

Figure 3: System Structure.

## Run Environment

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

## 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.

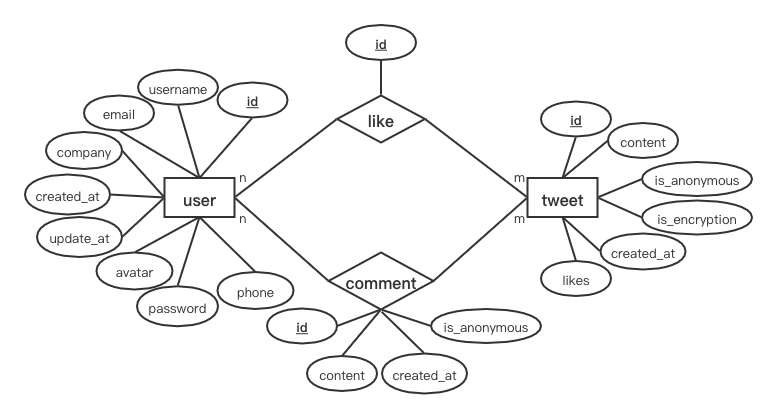


Figure 4: ER Diagram.

The relationship pattern of this project is:

1. **user** (**id**, username, email, company, avatar, password, phone, create time, update time)
2. **tweet** (**id**, content, anonymous, encrypted, created when, like)

Foreign key: user id, user name

1. **comment** (**id**, content, created when, anonymous or not)

Foreign key: user id, tweet id

1. **like** (**id**)

Foreign key: user id, tweet id

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

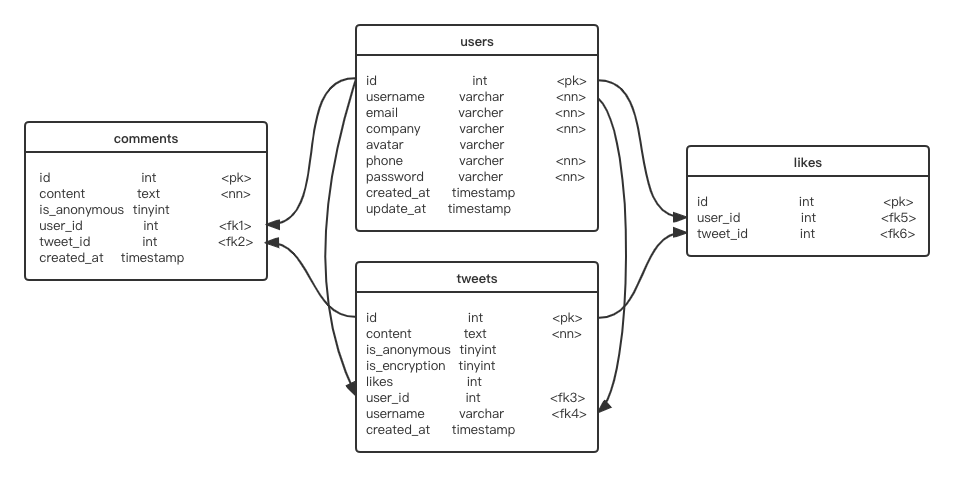


Figure 5: Database Design & Foreign Key.

## 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 log in, the system caches the data of the current user 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:** Sensitive information detection functions. Including sensitive information detection and NER entity recognition. Functions store as a Python file, called in this project by using python-shell.
5. **Exception handling:** If the input data is incorrect, the system throws an exception and prompts the user.

## Function Layer

All the specific data table designs are displayed in appendix. The overall design and foreign key relationship of the database are shown below.The functional layer is divided into six parts, including basic functions, send posts, read posts, decrypt posts, comment posts, search posts.

### Basic Functions

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

### Send Posts

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 post, 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 through the Base64 algorithm. 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 tweets, and the second part is entity detection using the NER tool.

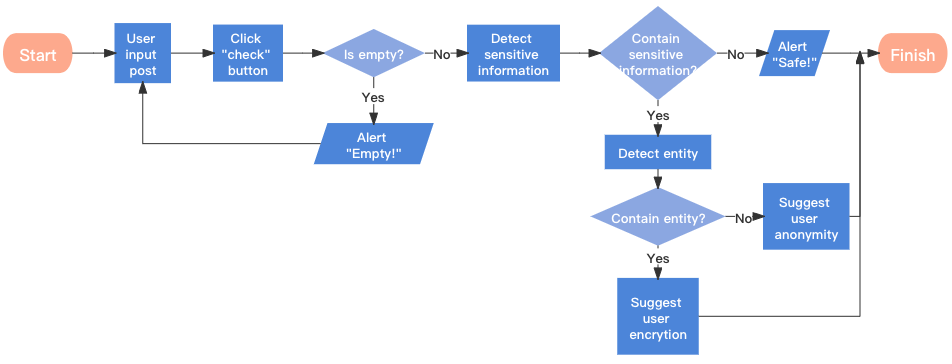


Figure 6: Sensitive information detection mechanism.

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 NER tool includes name, salary, location, and company name.

If the system detects inappropriate statements but does not detect any 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.

### Read Posts

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

### Decrypt Posts

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. Include:

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.

### Comment Posts

Users are free to comment on any readable post, add emoticons, and remain anonymous.

### Search Posts

Users can search for posts based on keywords.

## Display Layer and Front-end UI

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

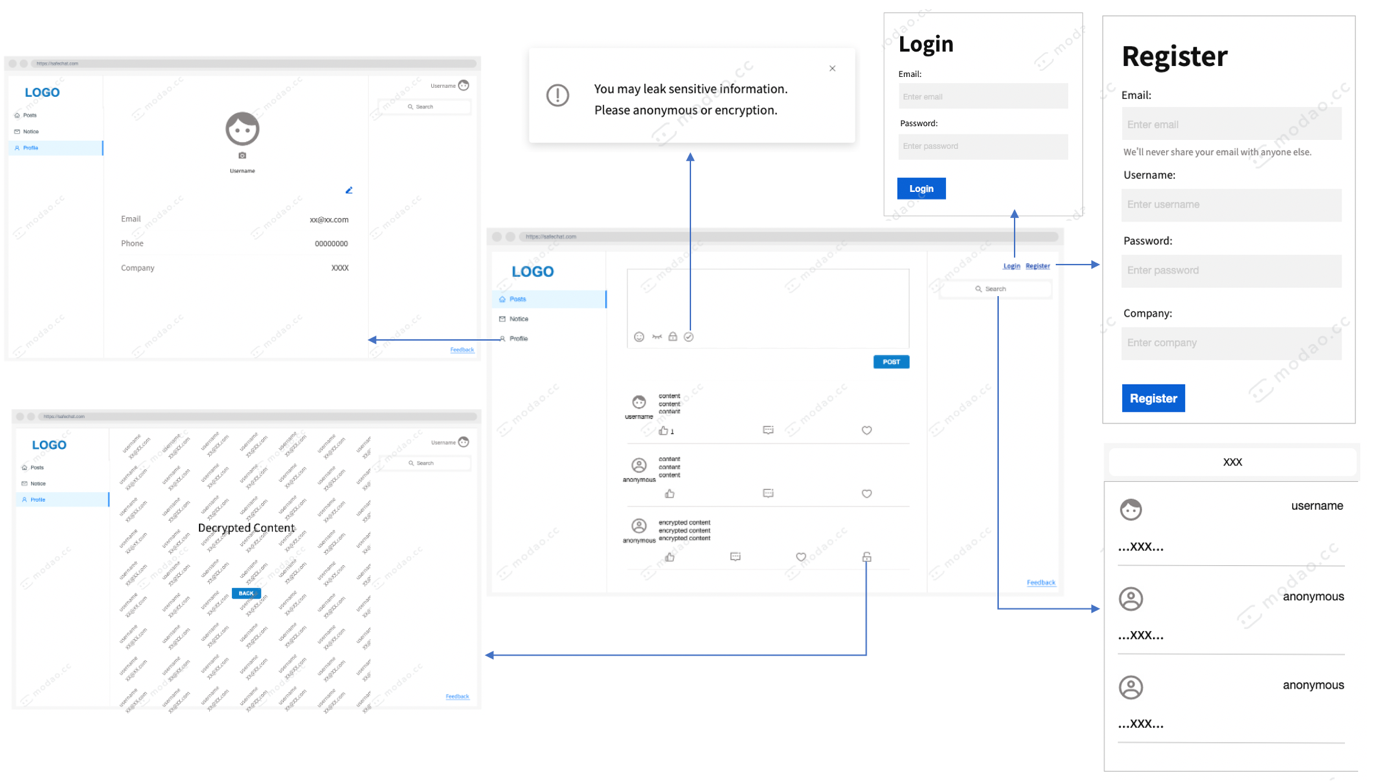


Figure 7: Main pages prototype.

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.

# Implementation

This chapter introduces the concrete implementation of the system. To help readers experience a clear logical structure and have a better reading experience, the introduction sequence of this chapter is the same as that of developing the system. The order is the run environment, database, functional layer and data processing layer.

## Database

Create a local MySQL database named "Safetweets". The tables are not created at the terminal, but via knex.js, which is an SQL constructor based on Node.js. The reason for using knex.js instead of creating tables directly on the terminal is to make the system more portable. 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.

## Function Layer

The front-end is realized by the bootstrapvue and the back-end is developed by Node.js. This chapter mainly introduces important parts of building as well as the challenges encountered and solutions. Introduce the modules following the sequence of function requirements: basic functions, send posts, read posts, decrypt posts, comment posts and search posts.

### Basic Functions

#### Encryption Password

The basic function part, including user login and register, so 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. Common hash encryption methods include MD5 encryption. However, the biggest problem of this algorithm is that there will be collisions, that is to say, different texts can get the same password. If the original text is M1, you only need another password with the same hash value to log in.

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. 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.

#### 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:

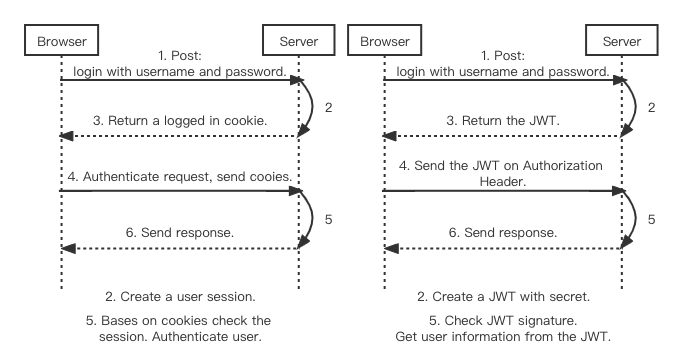


Figure 9: Session Authentication. Figure 10: Token Authentication.

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 server overhead. 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. Therefore, JWT is chosen as the cross-domain authentication scheme.

### Send Posts

Due to cannot find a suitable open-source rich text editor, I developed a text box. The text box contains four buttons for expression, anonymity, encryption and detection.

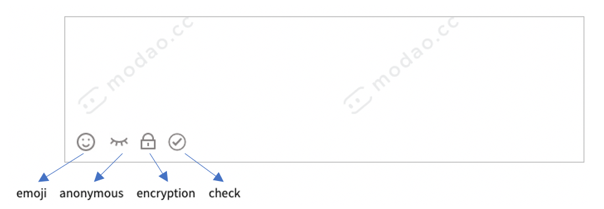


Figure 11: Editing box.

**Anonymous button:** Since jQuery is not available in Vue, use v-model instead for the two-way binding of data. By listening for user input events to update data, which improves Vue supports not only the MVC pattern but also the MVVM pattern. V-model also works well with text editors.

**Encryption button:** Since the encryption function is not to ensure that people cannot read the plaintext, but only transcoding. Therefore, a reversible encryption or encoding method is required. Based on this requirement, Base64 encoding is suitable in the encryption part.

**Check button:** Sensitive information detection includes sensitive content detection and NER.

#### Sensitive information detection mechanism

The purpose of this section is to detect sensitive information against text content; hence we consider the use of 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 tutors, searching materials and my own understanding, I chose the following keywords: "Job", "work", "overtime", "boss", "employer", "colleague", "workmate", "salary", "wage", "income", "burnout", "Equality", "get fired" and "get the sack".

1. Crawler Data

Crawler data by using Twitter's official API V2 and Postman, which is an API platform for building and using APIs. At first, sign up for a Twitter developer account to access to Twitter's API keys, API Secret keys and Bearer tokens and authenticate identities with that information. There are two endpoints for search, “search\_recent\_tweets” and “search\_all\_tweets”. The former searches only all tweets from the past seven days, while the latter begins with the first tweet in March 2006. For this project, to gain more information and train classifiers better, we chose to use “search\_all\_tweets”. 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. Input keywords into Postman and save the obtained .json file as .xlsx file for easy annotation.

1. 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 data and non-sensitive data. After manual annotation, total obtain 800 valid data. Then randomly select 650 valid data and store them in excel. It contains 300 sensitive data and 350 non-sensitive data as the final data set.

1. Data Preprocessing

Data preprocessing includes removing nonsense words, determining whether the string is in English, removing punctuation, part of speech restoration, and filtering stop words. This step relies heavily on the nltk library. Nltk, which stands for Natural Language Toolkit, is a general Python library for NLP research. Nltk can meet almost all data preprocessing requirements.

The removal of nonsense words mainly refers to removing the username follow "@", the tag "#" and the URL links that begin with "http". These are parts of the text that don't make any sense for analysis. The next step is to use the "enchant" dictionary to filter all pure English strings. Compared with word stem extraction, the result of word shape restoration is more readable. In terms of the processing of stop words, combine nltk's stop words table with the stop words table I set to delete all the stop words in the valid data. Next, according to sensitive and non-sensitive data, each processed tweet is spliced separately to generate two new data lists.

1. Feature Extraction

Carry out vectorization and feature extraction for all data by the sklearn library. Sklearn is a powerful Python machine learning library that covers everything from data vectorization 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. These two steps can also be combined into one function, TfidfVectorizer ().

1. Train Model

Usual text classification models include SVM, K proximity, Naive Bayes, decision tree, Adaboost and random forest.

**SVM:** SVM is a binary classification model. Its basic model is a linear classifier with the largest interval in the feature space. SVM contains different kernel functions to solve the problem of linear inseparability in real data. I use linear kernel and poly kernel to train the SVM model.

**K-NN:** The idea of the K-NN algorithm is that 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, when the number of neighbours is 13, it will have the maximum accuracy with the minimum computation amount.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Neighbors** | 5 | 7 | 9 | 11 | 13 | 15 |
| **Accuracy** | 68.99% | 73.64% | 72.09% | 76.74% | 77.52% | 77.52% |

Figure 13: Accuracy of different Neighbours.

**Naive Bayes:** is a simple probabilistic classifier based on Bayes' theorem. The naive Bayes dependence requires Laplace smoothing to deal with possible zero-probability problems, so set a number for the Laplace smoothing parameter. And since the sensitive information and non-sensitive information in the data set are not balanced, the prior probability needs to be considered by the Bayesian model.

**Decision tree:** is 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. And when setting the minimum number of samples required to 4 would obtain the highest accuracy.

**Adaboost:** is 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 stronger final classifier. Set the number of iterations of the parameter to 100. When the learning rate is 80%, it achieves the highest accuracy.

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

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** | 79.85% | 66.67% | 77.52% | 81.40% | 73.64% | 62.02% | 77.52% |
| **Precision** | 56.81% | 2.27% | 52.27% | 68.18% | 47.27% | 70.45% | 45.46% |
| **Recall** | 78.13% | 100% | 74.19% | 75.00% | 65.62% | 70.46% | 80% |
| **Time** | 0.03s | 0.04s | 0.02s | 0.00s | 0.00s | 0.03s | 0.03s |

Figure 14: Performance of different models.

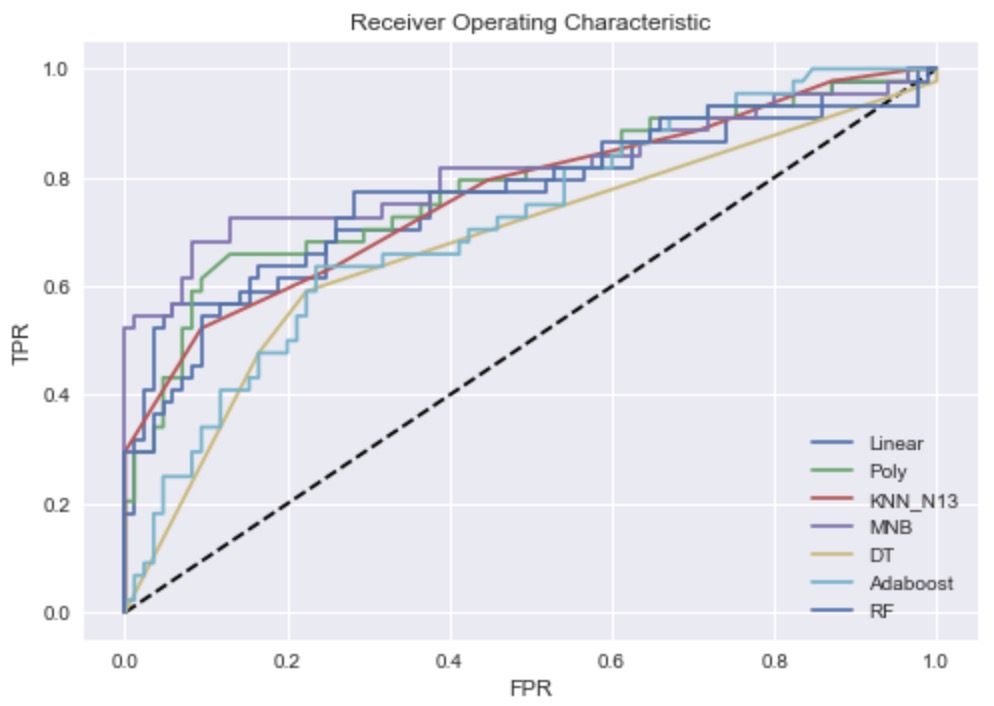


Figure 14: ROC curve of different models.

The results show that naive Bayes performs best in accuracy, precision and recall. In the ROC curve, naive Bayes also performs best. Therefore, adjust the parameters of naive Bayes continuously, and the model achieved the highest accuracy when the Laplacian smooth adjustment was set to 1.1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **Precision** | **Recall** | **Time** |
| **Naïve Bayes** | 83.72% | 68.18% | 81.08% | 0.00s |

**Figure 15:**  Performance of Naïve Bayes.

1. Save and Call Model

The system often needs to use the call model for sensitive information detection, so the model should be saved, when necessary, directly call out the use. Models are saved and loaded through the joblib library.

# Test and Evaluation

## Testing

The software testing uses Selenium Python to implement basic automated testing. Also need to use WebDriver, a third-party library for implementing web test automation. Selenium automatically manipulates browsers, such as 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 state of the application 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 sensitive information detection. In particular, it takes a long time to detect sensitive information, so you need 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 Functions, Send Posts, Read Posts, Like Posts, Decrypt Posts, Comment Posts and Search Posts. This is the number of test cases per module and the results. The detailed test is displayed in appendix.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Module | Test Cases | Assertions | Pass | Fail |
| Basic Function | 3 | 3 | 3 | 0 |
| Send Posts | 4 | 10 | 4 | 0 |
| Read Posts | 1 | 1 | 1 | 0 |
| Decrypt Posts | 2 | 3 | 2 | 0 |
| Comment Posts | 2 | 5 | 2 | 0 |
| Search Posts | 1 | 1 | 1 | 0 |

**Figure 16:**  Test result.

## User Evaluation

Testing only checks the system if there are undetected problems, but specific usability needs to be evaluated by real users. In social networking markets, the primary determinant of user activity 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. In this way, some problems that have not been noticed can be timely improved. However, user experience is a purely subjective psychological feeling, with many uncertainties and individual differences. It is not easy to accurately evaluate user experience. Therefore, to accurately evaluate the user experience, the evaluation criteria should focus on practicality and operability. All evaluation criteria should be quantifiable, measurable, observable and reproducible.

### 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. And 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.

### 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 the appendix. The second part is the questionnaire at the end of the task. The questionnaire is based on Brooke's testing table and the author made modifications according to the characteristics of SafeTweet. It is for analyzing user experience and satisfaction. The questionnaire is as follow.

#### 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.

### Result Summary

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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Duration of Evaluators Execute Tasks | | | | | Average Duration |
| E1 | E2 | E3 | E4 | E5 |
| A. Basic Function | | | | | | |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |
| B. Send Posts | | | | | | |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |
| C. Read Posts | | | | | | |
| 1 |  |  |  |  |  |  |
| D. Decrypt Posts | | | | | | |
| 1 |  |  |  |  |  |  |
| E. Comment Posts | | | | | | |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| F. Search Posts | | | | | | |
| 1 |  |  |  |  |  |  |

**Figure 17:**  Results of evaluation task.

|  |  |  |
| --- | --- | --- |
| ID | Question | Average Score |
|  |
| 1 | I think that I would like to use this system frequently |  |  |
| 2 | I do not think the system unnecessarily complex |  |  |
| 3 | I think the system is easy to use |  |  |
| 4 | I do not think that I would need the support of a technical person to be able to use this system |  |  |
| 5 | I think the various functions in this system are well integrated |  |  |
| 6 | I think there is no inconsistency in this system |  |  |
| 7 | I think that most people would learn to use this system very quickly |  |  |
| 8 | I do not think the system very cumbersome to use |  |  |
| 9 | I feel very confident using the system |  |  |
| 10 | I do not need to learn a lot of things before I could get going with this system |  |  |

**Figure 18:**  Partial results of the questionnaire.

# Conclusion

<Each new chapter should appear on a new page.>

Background concepts (if required) and overview of relevant previous work (critically evaluate strengths and weaknesses).

###### Requirement List

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Module** | **Description** | **MoSCoW** | **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 Posts | 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 | 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 Posts | 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 Posts | 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 Posts | 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 Posts | 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 |

###### Screenshots

###### Task List

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

###### Questionnaire

|  |  |  |
| --- | --- | --- |
| ID | Question | Score 1 - Strongly disagree 5 - Strongly Agree |
| 1 | I think that I would like to use this system frequently |  |
| 2 | I do not think the system unnecessarily complex |  |
| 3 | I think the system is easy to use |  |
| 4 | I do not think that I would need the support of a technical person to be able to use this system |  |
| 5 | I think the various functions in this system are well integrated |  |
| 6 | I think there is no inconsistency in this system |  |
| 7 | I think that most people would learn to use this system very quickly |  |
| 8 | I do not think the system very cumbersome to use |  |
| 9 | I feel very confident using the system |  |
| 10 | I do not need to learn a lot of things before I could get going with this system |  |
| 11 | I feel that the modules are not performing well in terms of usability, but performing well. a. Basic function b. Post c. Read d. Decryption e. Comment f. Search  Reason: | |
| 12 | I feel that the modules are not performing well in terms of learnability, but performing well. a. Basic function b. Post c. Read d. Decryption e. Comment f. Search  Reason: | |
| 13 | I feel that the modules are not performing well in terms of memorization, but performing well. a. Basic function b. Post c. Read d. Decryption e. Comment f. Search  Reason: | |
| 14 | I feel that the modules are not performing well in terms of efficiency, but performing well. a. Basic function b. Post c. Read d. Decryption e. Comment f. Search  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. Zheng F, Ma L, editors. A Multi-layered Friend Recommendation System on Twitter. 2021 The 13th International Conference on Computer Modeling and Simulation; 2021.

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

5. 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 &amp; Data Mining; London, United Kingdom: Association for Computing Machinery; 2018. p. 120–9.

6. 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.

7. 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.

8. 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.

9. 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.

10. 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.

11. 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.

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

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

14. 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.

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

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

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

18. Waters K. Prioritization using moscow. Agile Planning. 2009;12:31.