University of Kent | Computing

CHATAHOLIC

CO600 Project
Technical Report

KA KEI CHOI (KKC21), LUN LI (LL375), XIAOYAN ZHOU (XZ72)
SCHOOL OF COMPUTING
UNIVERSITY OF KENT

Chataholic: Web-Based Chat Application

Ka Kei Choi (kkc21), Lun Li (ll375), Xiaoyan Zhou (xz72)
The School of Computing
The University of Kent
Canterbury, United Kingdom
Supervised by Olaf Chitil (O.Chitil@kent.ac.uk)

Abstract

This report will focus on the systems development life cycle of 'Chataholic'. The aim of this project is to develop a real-time chatting web application that can pair users according to their personality test results. The application is developed using the React and nodeJS web frameworks. We proposed a simplified personality test and a simple pairing algorithm which are implemented in our pairing system. This project is evaluated by users using the demo application and answering some questionnaires over the quality of the system.

1. Introduction

Anonymous social application refers to the use of virtual accounts without forcing users to perform real-time authentication. This usually allows users to register for an anonymous identity to socialise. Users can freely express their opinions and engage in social activities. At the same time, due to anonymity, the mystery of social interaction stimulates people to continue to engage in it. Some popular dating applications such as Tinder, Tantan, and Gmu have introduced a way of getting potential friends based on their distance. In this project, we want to propose another approach to match users based on their personality.

1.1 Entertainment needs

In general, people get happiness from chatting with people who are sharing the same interests and personality. They can enjoy the time with ease and release their pressure. Activity in Chataholic is a completely anonymous process as no real identity information is required from the system. Users can create a "virtual person" with an exaggerated profile. Moreover, the users are passive matching others that are based on personality and are without any real-life image and social relationships. This leaves users with a curiosity. Through the process of exploring friends and chatting, the users' entertainment needs are satisfied.

1.2 Emotional Needs

For the youth, the great social pressure and the shrinking private space aggravate the dilemma that people have a lack of space to express their feelings. Chataholic, as a completely anonymous application, which enables users to get a sense of security and ease in chatting. Users can vent and express their inner anxiety and dissatisfaction to real life with less scruple. Therefore, to achieve catharsis and to relieve their pressure, Chataholic will be one of the options for users.

1.3 Anonymous needs

Users can use Chataholic's anonymous environment to get rid of the shackles of reality, establish an idealised image motive or show their hidden side. In Chataholic, users do not have a fixed name and their user id would not be shown on any page. Furthermore, user profiles can be modified at any time and the matching results can be canceled through corresponding mechanisms, thus users can be re-anonymised multiple times. As a result, users have more opportunities to get rid of the self-restraint of reality by hiding their social identity information.

1.4 Data Protection

We have not taken effective safeguards against our database because of lack of development time. This could lead to a massive data leak if our unprotected application was launched. For example, Uber data breach from 2016 affected 57 million riders and drivers. Uber paid $100,000 to hackers to get rid of the data in order to keep the breach under wraps.[12] Therefore, we will take the following protections to keep the database safe in further development:

1. Data Encryption: encrypting the stored and transmitted data so that the contents of the data are unknown to those who do not know the decryption algorithm.
2. Database Firewall: deploy a firewall system which takes active defense mechanisms to realise database access behavior control and high-risk behavior blocking.
3. Data masking: using a special data masking algorithm to deform, mask, randomise and transform sensitive data into fictitious data.

2. Background

2.1. Marketing

As we considered, a project like this should be related to reality, especially to find its market place and a future marketing plan. Actually, we have started our project with a further business blueprint. In this section, we will introduce our market placement from market strategy and platform strategy.

2.1.1. Market strategy

A market strategy is a long-time blueprint and is growing with the future market demand. For our project in this stage is to find our position. As the demands for qualitative conversation and company, efficient cooperation and true relationships are increasing more than ever, we come up with the idea of this project. The number of potential customers for our app has been increased rapidly since it aims to meet a general demand in the whole society. With a great customer base and the development of the further features of our app (e.g. More interesting tests for different aspects such as food, shopping and lifestyle preferences), we believe some user data such as the user’s shopping hobbies, food preferences and clothes styles could be collected and bring relevant businesses. For example, we can pop up shopping suggestions based on the testing results or even build up our shopping system, etc. In this case, further development with another demand cycle starts.

2.1.2. Platform strategy

A platform strategy is for us to consider 'How our application could serve users?'. As one of the main features of the application is communications, we are going to focus on the development of interaction functions between users. The ideal functions could be, for examples, mini games for two users and sharing cinema for a small group of users, etc. This could leave users with a sense of freshness and improve user experience. The further development could follow the market strategy which is based on users' preference information (the relevant businesses as mentioned).

2.2. Personality test

As mentioned before, our pairing system matches users based on the results of personality assessment measures. Though the research, we found a wide variety of personality scales and questionnaires that have been developed, including Comrey Personality Scales (CPS), DISC assessment, Minnesota Multiphasic Personality Inventory (MMPI), and Myers-Briggs Type Indicator (MBTI). MBTI is considered to be the best approach for our project. Since MBTI does not show the strength of the capability, it uses preferences to determine tendencies, not the ability strength nor the degree of ability and there is no right or wrong in the type indicator. More details of MBTI is explained in section 6.

2.3. Development model

When we are choosing the software process model, we decide to choose plan-driven processes------Waterfall. The core idea of the waterfall model is to simplify the problem according to the process, separate the implementation of the function from the design, and facilitate the division of labor and cooperation. Moreover, it separates the logical implementation from the physical implementation using a structured analysis and design method. The software life cycle of this project is categorised into six activities of planning, UI design, system design, functions implementation, function testing, and maintenance. These stipulate a fixed sequence of top-down and interconnected, like waterfalls, level by level whereabouts. The clear structure of the waterfall and generated documents make the progress visible. All of our planning documents were finished before the project started. The development progress is seriously following our quality assurance plan.

3. Aims

This project aims to build a real-time chatting web application that could match two users by using a simplified personality test and pairing algorithm. A database is needed to store account details, user profiles, friends list, and chat history. All the required functions are listed in the Requirement and the Specification files in the Corpus.

4. Development decision

4.1. Cross-platform development

Before starting the project, some essential background research and reading will be untaken to support us with a better understanding of Instant messaging systems. We also looked into some other

relevant projects and applications for inspiration and deeper understanding.

First of all, we start looking at the choice of an integrated development environment (IDE). At the very beginning, we were planning to use Android Studio to develop an application for the android platform only. It is free, and it has Gradle-based build support and a rich layout editor that allows users to drag-and-drop UI components. However, this decision will limit the customer to Android users only. After reading the develop log of some other IM apps, we changed our idea to design a cross-platform application. The Android studio is no longer sufficient for our cross-platform development. Therefore, a web-based application would be our first option. To build a web-based application, we don't need to learn additional programming languages and this will reduce the development time. As a final decision, we use "IntelliJ IDEA" as our IDE and we need a web application framework that supports web services and web APIs.

4.2. API and frameworks

This section discusses the decision choice of Application Programme Interface (API) and the framework we used for our project. As the application would be implemented on all platforms including Windows, Android, IOS, etc., an API is going to be used for accelerating the development. By considering the developing period, human resources and skills we have, the 'HTTP' is the easiest form of API that is going to be used and brings us the choice of the framework for the web service – React. The following would be the detailed comparison and reason for us to make these choices.

4.2.1. API

There were limited choices for us to manage the cross platform during development. In general, to develop an application that can run on different platforms, multiple platform-specific versions would be required and they would share the same structure and logic. However, this approach would be a huge project for a small group to develop within four months as we need to overcome the programming languages, different developing tools and massive tests. Therefore, an API will be a solution for this situation. API is a set of routines, protocols, and tools for building a software application [8]. Basically, with an API, we can connect the components on different platforms to a central server, although an application interface is still required, the amount of work would be decreased and make the development to be delivered in time become possible.

There are lots of API choices for us to manage the connection of the components and we can also design our own interface for the application. However, as we have a lack of time and skills, 'HTTP' becomes the first choice to be considered. Firstly, as a web-based application, what we actually need to build is a web page which can be visited by different devices, only a few optimizations are required. Secondly, we are more familiar with web development languages such as JavaScript, PHP, etc. Finally, there are similar applications and this strategy has been successfully used several years ago by the earlier Facebook

application. In addition, plenty of libraries could be found on the public internet.
In conclusion, for the API choice, an advanced framework would need to be chosen, and so 'HTTP' would be the most realistic way for us to implement all required functions in time. This is also the easiest way to develop an application like this project.

4.2.2. Framework

The framework is required as we decide to develop a web-based software. A web application framework is a type of framework or foundation that is specifically designed to help developers build web applications [10]. In other words, a web framework is the foundation of an application and a good framework choice is important as it influences the difficulty, flexibility, and stability of the development.

There are lots of choices for the frameworks. We have researched on some common frameworks and picked three popular frameworks to do a comparison. These three frameworks are picked: Angular, React and CodeIgniter. Only one would be chosen as our framework.

Angular. Angular is a front-end framework with a component-based architecture framework. The mainly used tool language is JavaScript and Typescript. With a large size of the framework, there are many functions that can be used on our project. But the large size could also be a problem and extend the learning period.

React. React is not only a framework but also a large library. It adopts the component-based architecture which leads to a small size of the framework. The mainly used language tool is JS. The large library design could build up the lack of plenty of functions in Angular. Besides, React also takes advantage of loading speed and maintenance.[9]

CodeIgniter. CodeIgniter is a powerful PHP based framework with a very small footprint, its small size gives it convenience, loading speed, and even zero configuration, the lack of library becomes the fatal weakness. However, we do have a better understanding of this framework since it was used in one of our projects.

As we aim to develop a chat app., this project required instant communication, good connection with the database and server. Therefore, a large library and functions support are necessary.
PHP-based may create difficulty for the group member to implement although we have a basic understanding of it.
Therefore, CodeIgniter would not be considered as our framework. The comparison between Angular and React was difficult to make, as both of them have good libraries and functions. Furthermore, they are JavaScript-based which is more convenient for us. However, due to the time limit, the learning and implementation period become the main factors for decision making. As React has a smaller size and a shorter learning period than Angular, it has been chosen as our framework.

5. Architecture

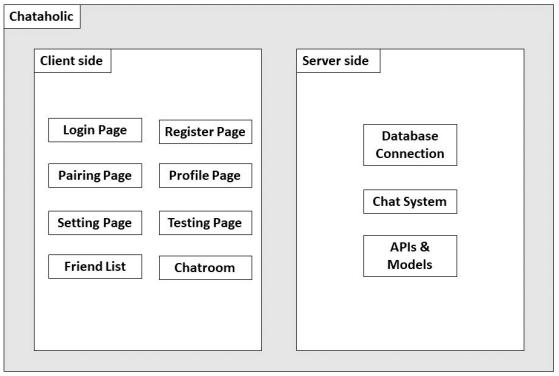

Fig 1. Chataholic system architecture

5.1 System design

The system is split into two sections: client side and server side. As shown in Fig 1, each small part in the client side represents a React component which main purpose is to provide a fine and functional user interface. There are details of the UI that have been demonstrated in our video. On the other hand, there is a pairing system implemented in the testing page and pairing page. This system aims to achieve personality testing and matching users. More details of the pairing system are explained in section 6.

The main task of the server side is to handle MySQL database connection, handle the communication between client side and server side via WebSocket and perform a query to the database that is requested from the client side. There are more details on the Chat System as explained in section 7.

6. Pairing system

This section introduces our test rules and pairing rules that used to achieve matching users based on their personality. We also discuss the theory behind the rules and how we design and simplify the testing and pairing algorithms.

6.1. Myers-Briggs Type Indicator

The Myers-Briggs Type Indicator (MBTI) is to differ psychological preferences in how people perceive the world and make decisions with a series of questions [3]. The test was based on the psychological type theory by Carl Jung who proposed two dichotomous pairs of cognitive functions: the rational functions: thinking and feeling and the irrational functions: sensation and intuition [5]. And by collecting information and calculations, the two pairs of cognitive functions which refer to people's personalities could be represent with four pair of letters and leads to 16 personalities in total [5]:

- Outwardly or inwardly:
  Extraversion (E) or Introversion (I)
- Reality or intuition:
  Sensing (S) or Intuition (N)
- The way you make decision:
  Thinking (T) or Feeling(F)
- The way you live your outer life:
  Judging (J) or Perceiving (P)

The four dimensions of the MBTI test could Digitize and give the general idea of a person's personality, and the test itself has been widely used in business. We believe that the test can help us determine our user's personality generally and there is different interaction between different personalities. This section is aimed to give an overview of the testing and pairing algorithm used in our application Chataholic and how we simplify it for users, which is also designed based on the

MBTI test.

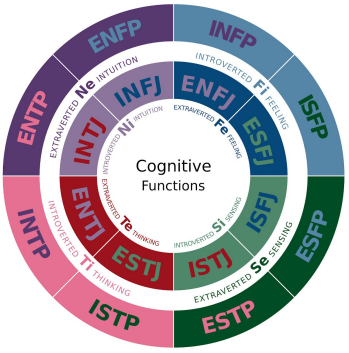

Fig 2. A diagram depicting the cognitive functions of each type. (Author of the referenced image: JakeBeech)

6.2. MBTI Test algorithm design

There are several calculation models found online that are related to our work, the core of most of them are turning the users' choices into scores and different problems are mapping to different latitudes. There are several examples we could learn to design and build our rules.

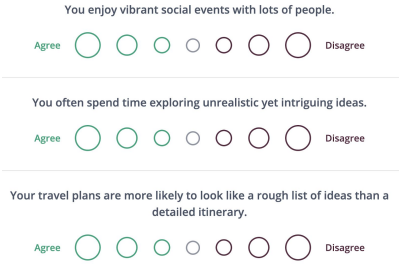

Fig 3. MBTI calculator model 1 (source: 16personalities.com)

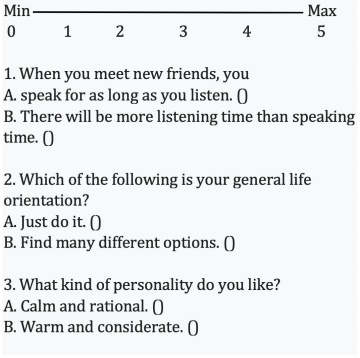

Fig 4. MBTI calculator model 2 (source: MBTI calculator)

The questions in Fig 3 are designed into how agree/disagree you think of the question and each level mapping to a different score. On the other hand, the questions in Fig 4 are designed into A or B questions however you need to give a score to A and B which total would be five. And the score would be used to calculate the final score for each latitude. The model itself is more subjective.

The difficulty for the part is to find out how the formulas actually work and simplify it to fit our online users which means the testing itself should be much easier and shorter to take. The solutions would be discussed in the following part.

6.3 Test rule design and simplification

6.3.1 Calcification

As mentioned before, there are four pairs of psychological functions (PF) each associated with 2 preferences, which are behavior function-{E, I}, perceiving function-{S, N}, judging function-{T, F}and a function for worldview-{J, P}[5]. We notice that it means there would be four pairs of opposite choices for a person to reflect his/her personality and each opposite pair is based on the information collected from the questions. In this case, we considered it as a decision tree classification as shown in Fig 5.

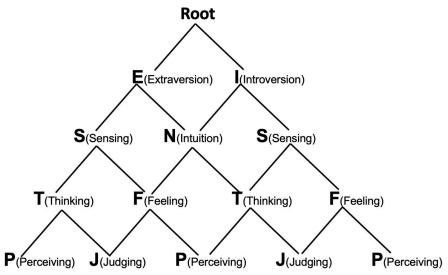

Fig 5. A decision tree that shows how the classifications happen in the MBTI.

And for each function, analyzing it as an overview, there would be subjective questions based on human behavior for users to answer and each question could reflect a function. In this case, what we need to do is to calculate the questions. For example, question "You enjoy vibrant social events with lots of people." could be related to *E* and *I* [6]. And with the help of the MBTI Manual [6], we sorted out a question list and which function it related to. Finally, we select five questions for each function (20 questions in total from 98 questions) which leads to a balance between accuracy and the users' experience (easy and fast to finish, avoiding long time struggle) then we mixed them up to prevent self-suggestion during the testing.

6.3.2 Calculation

As mentioned above, there several models we could learn and we chose the pick-one-of-two model to meet the idea of classification (decision tree) and simplify it which is also an earliest calculation model in the MBTI Manual [6]. As in manual, for one question, there would be two choices A and B (see Fig. 3), for each choice you need to give a mark in total as five to define how does the description close to yourself, for example, I give A three marks then it would leave two marks for B. There would be even questions for a function which avoids the same mark for A and B. A simplification is made here since A is given three marks, B' s two marks would lead to a final value of A, as well as B.

In our rule, the choice of A/B is mapped to a specific pair of preferences in a PF. The choice of A/B would increment the score to the preference it associated with. For example, A is mapped to *E* and B is mapped to *I* in a behaviour function question. Choosing A would increment the score for *E*, otherwise increment *I*'s score. As a result, by comparing the score of the preferences in a particular PF, we can get a specific preference of that PF.

6.4 Pairing Rule

The pairing rule happens in the vertical part which is about the interactions between different personalities. It could be much more complex because of the flexibility of a human's mood and lifestyle. In this section, we would give some general rules for the general behavior of each personality. For example, An *E* type could be a great talker with strong opinions, he/she could be suitable for an *I* type who is shy but wants someone to talk to, but at some point, an *E* type could be offensive to an *I* type. As a group of computer science students instead of psychology, this could be inaccurate.

For the pairing algorithm, it is designed based on the personality characteristics for each personality described in [1]. For example, the description for a ISTJ type is: responsible, organizing, efficient and

practical, take things seriously, traditional and conforming ,etc. he or she could be suitable with a person with ESTJ type which is described as a great talker, dependable and responsible, find comfort in family routines and traditions, etc. as they are all comfortable with traditions and take things seriously.

For this stage, we find that, according to the personality characteristics for each personality described in [1], most personalities defined their same personalities as their neutral personality which could be considered as a personality could easily cultivate feelings and get good impressions instead of having good impressions at the first time. To guarantee the basic user experience right now, we strictly pairing the users with the same personalities.

There are a lot of parts that need to be considered during the pair system, but at the current stage we only consider the similarity of two personalities. Later we would consider the complementarity between different personalities. This part would be one of our further topics to work on.

7. Chat system

This section introduces our chat system that is used to achieve real-time messaging.

7.1. Structure

The chat system is divided into two parts: client-side and server-side. As Chataholic is a web-based app, we use WebSocket to communicate between both sides. As WebSocket is widely used across different applications, for example, live updating maps and geolocation, streaming stock quotes, etc. There are rich resources and libraries for us to develop our chat system easily. First of all, we would talk about some details of the WebSocket, then the main functions and the structure of both sides.

7.2. WebSocket

WebSocket is a computer communication protocol that provides a bidirectional and full-duplex communications channel that operates over HTTP through a single TCP socket connection.[7] By using WebSocket, we can achieve our project main goal, real-time messaging. In addition, most of the web browsers support the WebSocket including Google Chrome, Safari, Internet Explorer, Microsoft Edge, Firefox, etc. This allows most of the users to communicate from different devices and platforms as shown in Fig 6.

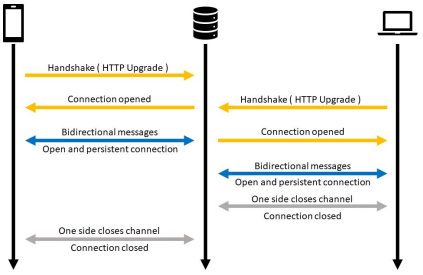

Fig 6. WebSocket visual representation

7.3. Client Side

On the client-side, the code for the chat system mainly focuses on the listening and emitting events to the server-side through the socket connection. All the codes are located in the “socket.js”, main functions including emit connect and disconnect event, the chat message and history events.

7.4. Server Side

The server-side chat system consists of three parts: Handlers, RoomManager and Chatroom.

Handlers. The handlers handle all the events when receiving a message or signal from the client-side. It performs different functions according to the message heading. Main functions include handling the chatroom creation and deletion, sending messages, socket connect and disconnect events.

RoomManager. The roomManager holds a list of chatrooms and the main role is to manage these chatrooms. Everytime a user is connected to the socket, the user-related chatrooms are added to this list. On the other hand, the chatroom will be removed from the list when all the chatroom-related users are disconnected.

ChatRoom. The chatroom holds a list of chatroom-related users who connected to the socket, chat history and its own room id. Main functions include get its own chat history from the database, modify the chat history and the users list, broadcast message, etc.

In general, the flow of event would be (Server side = Ss, Client side = Cs) :

1. User login and emit an initial event to the Ss from Cs
2. The Ss responds to the event and adds the related chatrooms to the list. The system also adds the user socket to the room member list
3. The Ss emit the chat history event to the Cs
4. The Cs receive the chat history and display it on the chatroom page
5. The Cs emit a new message event when user sends a new message in the chatroom
6. The Ss response to the event by broadcasting the new message to all the chatroom members
7. The Cs receive the new message and display it on the chatroom page
8. Repeat 5-7
9. The Cs emit a disconnect event when user log out or close the app
10. The Ss responds to the event by removing the user from every related chatroom. If the chatroom member list is empty, the chatroom also remove from the chatroom list

We have tested the system on different browsers and devices under a local network environment. As a result, this project successfully achieved real time messaging by using WebSocket.

8. Test

8.1 Functional Testing

We use black-box testing based on the user cases that we designed in the early stages. We do a simple test for each function during the project development process. Every time when a new function is added, we will run tests to ensure there is no conflict with the previous code. Theoretically, the black box test can find out all the errors in the program only if it uses an exhaustive input test and considers all possible inputs as test conditions. In fact, there are infinite testing situations. People should not only test all legitimate inputs but also test those illegal but possible inputs. In this way, complete

testing is impossible, so we need to carry out targeted testing. All the main functions have been tested and recorded in the Test result file in the corpus. We try to test as many test cases as possible in a limited time, and all the tests have passed as expected.

8.2 User testing

User testing is the final stage of our developing period. As Chataholic is a social media application, user experience is very important in our project. We need to know how users feel about our system in order to improve our work in the future. A total of 20 students were invited to use our demo application which is run under a local network environment. The system will be evaluated by users answering some questionnaires on a 1 (strongly disagree) to 5 (strongly agree) scale. The questionnaire template is shown in Table 1. Each user's feedback has been recorded in the UserFeedback file in the corpus.

9. Evaluation

In this section, we will evaluate the test result that was taken by the user. Some of the user feedback gave us some fresh ideas that we didn't consider before and we can know if it is easy for users to use the app without guidance.

9.1. General System

The basic functions, including register, login, logout, amend friend list and user's profile, have all passed the testing during the developing period and get mostly positive feedback from the users. Most of them comment on the nice register/login page, easy and convenient authentication and validation and high completion of the profile system. However, some users suggested adding a function for direct login to the new account after registration. There is also an interesting discussion on the user icon, that some users approve of the same icon for everyone which shows respect and mystery for their privacy, but others think it's too drab for the app. This could be a good point for our further development to balance novel and privacy. Overall, the basic system has met all requirements and achieved success in both design and implementation. Advice has been taken for further improvement.

9.2. Chat system

The chat system has a high weight in this project so it would be evaluated as a single part. We have several goals for it: real and non-real time messaging, keeping or downloading chat history and clearing chat history. In testing, users are able to achieve real and non-real time chat via WebSocket. Moreover, messages are uploaded to the database to be stored and be retrieved when the user enters the chat room. However, clearing the chat history by one user will end up the whole chat history of that specific chatroom being removed because the history is stored in one table. Another chatroom member will also lose his chat history as well. This is a large disadvantage of the system. Another negative point is that we haven't deployed the application to the online server. In this case, the chat system could only be tested under a local network.

User's feedback on this chat system provides some useful foresights including good implementation of the online and

offline chat system, useful history keeping and downloading, no emoji and picture sending, no notifications and the incomplete history deletion. All of these could lead to inconvenience for user experience.

For this chat system, we have missed several important functions especially notification and clearing history. Although some of the functions are not listed in the requirements, some user feedback reminded us that further work and effort need to be done.

9.3. Testing and Pairing System

The testing and pairing system is another important part of this project. As we are not psychology students, there are difficulties when we develop this system and we still cannot guarantee the accuracy of the personality test and pairing. The implemented functions for this system include testing algorithms, upload and update the testing result in the user profile, pairing fitted users.

The user experiences in this section are particularly important because the test result is based on the user's choices. We focused on the feedback from people who are friends and they are asked to do an additional official MBTI test with forty-eight questions. The result shows that 45% (9 out of 20) could get the same personality in both tests, most of them would only get 1-2 same personality preferences in the result. One of the reasons for this is the amount of questions. Lots of evidence has shown more questions could lead to a more accurate result, although our test version took a much shorter time to complete. This informed us more psychological factors need to be taken into consideration.

Our pairing rules followed the advice online which is much reliable. But the accuracy of our testing algorithm may impact on the user experience. There is some positive feedback on: relatively short testing time, direct enter to the new chatroom when finishing pairing, etc. However, there are high uncertainties in the pairing and testing. This is our most important part in the further development. Overview, the functions in this part have been done but there are unexpected performances in the system.

9.4. Cross-platform performance

As our project is a web-based application, it is able to run on different devices with web browsers. Although Chataholic only works under the local network environment for now, user feedback shows they are surprised the application could be run on their phones. The user interface can perfectly fit any screen size as we designed it as a scalable interface. Users mainly suggested that a terminal app could be implemented so that the web app could have a better performance on the phone platform. In this section, the performance of our application has met the expectation.

10. Conclusion

10.1. Overview

We developed a web-based social media application with a real-time chat system. We also offer a simple pairing system that is based on the user's personality. Overall, we think Chataholic is successful as it

achieves most of the requirements. However, we still have more to work on.

10.2. Future work

10.2.1. Chat system improvement

First of all, the chat system works well as users can chat in real time. However, each chatroom only contains two users. An advanced group chat could be made on this system. Furthermore, we would like to develop the notification function in the future. So Chataholic can notify users just like most of the mobile application does. The bug of the clear history that was mentioned before also needs to be fixed. Some messaging related features that we also want to add to our system including send images, file transfer, video chat, voice chat and turn on/off read receipts, etc. With these functions, our application will become more competitive.

10.2.2 Pairing system improvement

For the testing and pairing algorithm design, it is not difficult for us to work on the code. However, we lack knowledge in the field of psychology to design an advanced algorithm with high accuracy. Instead, we simplify the questionnaire for our simple web-based application and try to provide a better experience for users. Although the pairing system is not as great as what we expected, the pairing result is still appreciated by users. This system would be our further topic to work on.

Acknowledgment

We would like to thank Olaf Chitil for all the help for this project and to all who participated in our system testing.

Reference

[1] Charles Martin, "Looking at Type: The Fundamentals" Ph.D. Center for Applications of Psychological Type, Gainesville, FL 1997.[Accessed 3 March 2020]

[2] Center for Applications of Psychological Type. "The Story of Isabel Briggs Myers". Retrieved 2017-03-29. Available from: https://www.capt.org/mbti-assessment/isabel-myers.htm?bhcp=1 [Accessed 2 March 2020].

[3] The Myers-Briggs Foundation, "MBTI basics", 2014, Retrieved 18 June 2014. Available from: https://www.myersbriggs.org/my-mbti-personality-type/mbti-basics/home.htm?bhcp=1 [Accessed 1 March 2020].

[4] University of Florida George A. Smathers Libraries, Department of Special and Area Studies Collections, Gainesville, FL. 2003. "Guide to the Isabel Briggs Myers Papers 1885–1992". Retrieved 2005-12-05. Available from: http://web.uflib.ufl.edu/spec/manuscript/guides/Myers.htm [Accessed 2 March 2020].

[5] Myers, Isabel Briggs and Peter B. Myers (1995), Gifts Differing: Understanding Personality Type. Mountain View, CA: Davies-Black Publishing. ISBN 978-0-89106-074-1) [1980]. [Accessed 1 March 2020].

[6] Myers, Isabel Briggs; McCaulley Mary H.; Quenk, Naomi L.; Hammer, Allen L. (1998). MBTI Manual (A guide to the development and use of the Myers Briggs type

[7] indicator). Consulting Psychologists Press; 3rd ed edition. [Accessed 4 March 2020].
[8] [Online] RFC 6455 - The WebSocket Protocol - IETF Tools**.** Available: https://tools.ietf.org/html/rfc6455
[9] Beal V. (2019). 'API – application program interface', *weboopedia,* Available from: https://www.webopedia.com/TERM/A/API.html [Accessed 11 December 2019].
[10] Goel A. (2019). 'Top 10 Web Development Frameworks', *hackr.io,* Available from: https://hackr.io/blog/top-10-web-development-frameworks-in-2019 [Accessed 11 December 2019].
[11] Multiple (wiki). 'Web application framework'. *Docforge*. Archived on 2015-07-23, Available from: https://web.archive.org/web/20150723163302/http:/docforge.com/wiki/Web_application_framework [Accessed 11 December 2019].
[12] [Online] Darrell E. "Uber data breach from 2016 affected 57 million riders and drivers". Available: https://techcrunch.com/2017/11/21/uber-data-breach-from-2016-affected-57-million-riders-and-drivers/?guccounter=1 [Accessed 08 April 2020].

Appendices

Table 1. User Test Template

User Feedbacks	
Questions	**Score**
Does the general system function well?	
Easy to use without a guideline?	
Does the Login/Register page function well?	
Does the friend list page function well?	
Does the profile page function well?	
Does the chat room page function well?	
Does the chat system function well?	
Does the simplified MBTI test perform successfully?	
Is this personality test result you expected?	
Does the pairing system function well?	
Is this matching result you expected?	
Do you find the UI layout is reasonable and easy to use?	
Are you willing to pay for more application features in the future?	
Would you still be willing to use our app if we required a real-name authentication?	
Do you think Chataholic secure your privacy?	
Comment: **Suggestion on any improvement:**	