Messaging System Design Based on Using Servers and Encoding System

https://doi.org/10.3991/ijim.v14i10.15189

Musbah J Aqel (✉), Omar A Naqshbandi, Munsif Sokiyna, Pertsevoi Valentyn
Cyprus International University, Nicosia, Cyprus,
maqel@ciu.edu.tr

Abstract—Now, people all over the world use messengers to communicate with their families and employees at work, schedule meetings, congratulate on holidays, and conduct banking operations using bots built into messengers. However, it is challenging to improve such architectures, as a result of which WhatsApp still does not have Chabot's, whereas their competitors from Telegram already have them built-in. The essence of this project is to create a messenger that has an optimal ratio of functionality and code processing speed, as well as the ability to improve the architecture without high resource costs quickly. The basic version has features such as registration, logging in with your account, checking whether the user is online, messaging, and hash encoding for secure messaging over the Internet. The result showed a high potential for improving the application architecture in any part of the code within the acceptable level of influence on its other functions, as well as a more stable operation of the application itself in comparison with applications built based on Android or JavaScript. However, it was not possible to achieve a higher code processing speed in comparison with other Android or web applications. A method is designed and implemented to enhance the speed of code processing. The results produced by this proposed method and design has shown enhancement in the speed of code processing for messengers. Compared with other Android or with other web applications messaging systems.

Keywords—Java, MySQL, Threads, Hibernate, JDBC

1 Introduction

Once upon a time, America Online was in one faraway country. Furthermore, it had an amazing private Internet behind the fence, where instead of URLs there were "keywords": something in between the web page address and the purchased keyword in the advertisement. Companies fought for new keywords, as they are now fighting for domains, and the advertisement looked like this:" visit us on the world wide web at www.example.com, or type AOL Keyword: 'banking'". History has the property of repeating itself. Now the leading messengers play the role of America Online: they are all behind fences, incompatible with each other, everyone invents their keywords, they want to grab the user and never let go. Companies are not interested in openness: more significant players do not want to share users with smaller ones, much less become open. As a result, it is impossible to send a message even from WhatsApp to Facebook Messenger, even though both belong to the same company. Moreover, users value reliability and convenience above general openness, although it is annoying to many that some friends, for example, on Telegram, some on WhatsApp, and parents on Skype. However, the role of the open Internet, unfortunately, today no one plays. Since it is not profitable. The Internet is all based on the principle of the address - "domains" is kind of a street address, only on the Internet, for example, “www.google.com”. Knowing him, you will be able to get there, not knowing will long roam the network in search of the desired page. Messengers also work about the same way, they are a kind of closed clubs in which without knowing the address of another person, and there is no way to contact them. The messengers are of two types: Open source (Line, Riot.im and Signal) and closed source (Viber, WhatsApp and Telegram). Open source as the name suggests such applications have open source code where everyone can see what is happening inside the program code. While closed source does not allow. This what is happening inside, only the developers of this application know. In this work, an open-source application will be developed that is entirely ready for use. It will have a set of functions such as messaging, checking whether a user is online, message encoding[1], and account management functions such as registration and log in[2]–[4].

2 Problem Statement

Initially, instant messengers were created either as chats, for example, WhatsApp or as an application for calls - Skype, Viber. Later, messengers began to add functions that were not originally there. So, WhatsApp added audio call features, then video. Next came open APIs, bots, masks, statuses, payment methods, public channels. However, it is difficult to introduce new functionality or change the structure when the messenger has millions of users. In the same WhatsApp, there are still no APIs and bots. The main difficulty in creating an application for sending messages is development and architecture. The application structure needs to be developed in such a way that new features can be safely added to it.

3 Aim of The Work

The project’s target is to design an application that provides the opportunity for message exchange between people in real-time, to check the availability of users online, as well as adding new users to contacts for subsequent sending messages. To start working with the application, the user must first register and then log in to your account, fill out the contact list using the function to add new entries, after which it will be possible to start message exchange.

4 Significance of The Work

The main problem of such a project is to find the difference between different implementations of the messaging system; some of them are faster than others. On the other hand, others are slower, because of the amount of code. In this project, a comparison between different messaging systems and methods will be carried out and find the most balanced in terms of execution speed and amount of functions, that will help to select the best one that could be used with the messaging system.

5 Related Work

The new email has become Messengers. Now they are on fame high. Many online stores have long known this, and they use messengers for reviews and sales. Messengers have become the primary sales channel for many online stores. However, dealing with a vast number of requests in the traditional messenger system is extremely cumbersome. The question just grows harder when there are several messengers. The approaches mentioned below can be used to maximize the task performance, simplify the code and enhance the architecture of the interface, which in turn allows you to handle multiple requests.

5.1 Instant messaging service on android smartphones and personal computers[5]

The architecture suggested is a server module and client program that can implement the following steps:

1. First, the server program runs on the server machine.
2. Then the client-application program runs on an android mobile device and sends asking to attach with the server.
3. Once the consumer is with success connected, the server broadcast the list of all different active users to the client-application.
4. Client-application will read the list of all active users and might communicate with them.

This article presents an idea to create a service for network users, which will be deployed on the network server of any company that allows smartphone users to send and receive messages in a company free of charge. Such contact should not work with or take no knowledge arrangement with a mobile service provider. So, in this manner, it reduces the value of communication. It will increase the communication between numerous devices which provides compatibility with the private Computers with the assistance of Blue stacks that provides an interface between the user and also the personal computers or tablets to produce an economically-efficient communication by increasing its performance.

5.2 SQL injection detection for web applications based on elastic-pooling CNN[6]

Approximately 4.48 million real weblogs, including approximately 1.28 million SQL injection attack logs and 3 million standard logs are in use in the production environment. All logs are URL records with query statements. Standard and SQL injection logs used. Two hundred thousand samples (approximately 100,000 positive and negative samples each) were randomly selected for training purposes, and the rest were used as evaluation sets. There were no crossroads between the test and the training sets. This shows that the machine has been able to show a high level of adaptability and performance after some training. The results of the tests were higher, with each passing training than before.

5.3 A SQL injection detection method based on adaptive deep forest[7]

There are two positive examples in the tests, one aspect of which is provided by the DB and WooYun vulnerability submission tool. The other portion is bought within one month from running the SQL map and its modifying commands. The total number of recorded SQL injection samples is 10,000. Estimated 10,000 negative and 10,000 positive samples are included in the dataset. An SQL injection detection technique based on the deep-forest adaptive model was suggested in this article. Using ADF, templates can automatically be modified to boost the identification accuracy during the training process. Precision, alert and f1 scoring were contrasted between the proposed method and the traditional method of machine learning. The results show that higher efficiency was obtained.

5.4 Practical guidelines for boosting java server performance[1]

This document shows a way to avoid the following performance issues, thus simultaneously reducing the Java resource footprint. The paper also reveals that changes in functionality should increase the risk of adding vulnerabilities into the Java system because they only function if the software fulfils the preconditions. As an example, a few of the indicators turn a stable thread into a protected thread type. The code can also be open and maintainable. Therefore, the code’s textual ability can be improved.

5.5 Exception-chain analysis: Revealing exception handling architecture in java server applications [8]

The Handler Analysis research tests all capture clauses immediately in order to find out how the related conditions and details are used. Research by regulators reviewing realistic catch clauses in order to identify trap clauses to remove derogations. This data is paired with e-c connections that have been established by an existing static analysis and form e-C chains at the time of compilation without overhead runtime. A map of the e-c chains demonstrates the device recovery application architecture at several granularity levels: part, packet, class.

5.6 Request dispatching algorithms for web server ousters based on load balancing's[9]

This paper introduces the Web server cluster system architecture and two parameters for performance evaluation. The LTI algorithm is then developed to use the increment of the total response time as the optimization goal and is showed to be locally optimal. A sample test is used to demonstrate the efficiency of the algorithm. It is implemented using multi-thread technology. Each network connection is served by one worker thread. Every time a worker thread finishes transmitting a packet, it adds the length of the data packet to STT[i].

5.7 JPBC: Java pairing based cryptography[10]

JPBC was provided in this research as a Java port from the PBC database. For a non-cryptographer, JPBC offers the full interfaces and class environment for simplifying the use of bilinear map 7. JPBC is designed for the mobile World, support various forms of elliptical curves. JPBC also facilitates preprocessing, which can significantly improve measurement. First, the Setup algorithm has been implemented. It is used to generate system parameters and install encryption system into an application. Secondly, the Keygen algorithm has been implemented.

It is used to generate a secret key that will be used later. Thirdly, Sign algorithm that is used to mark a system to pair it with others. Fourthly, verify algorithm it takes the key that was generated by the Keygen algorithm and signature from Sign algorithm to validate the system.

5.8 Data encryption and decryption using RSA algorithm in a network environment[11]

The paper has presented encoding and decoding in a network that was with success created. With this software, information may be transferred from one pc terminal to another—to a different via an unsecured network. An eavesdropper that breaks into the message will return a meaningless message.

5.9 JDBC checker: A static analysis tool for SQL/JDBC applications[12]

JDBC Checker is a practical software application for Java / JDBC structures detecting code errors. Within Java, the JDBC Checker uses string search within for FSA programming, and successively utilizes the Soot System to check object files and to measure interprocedurally control-flow graphs. Tested on several codebases, including student team projects from a software engineering graduate course, demo content from web-based guides and data from other programs. In these systems, the computer has found known and unknown defects.

5.10 Towards NoSQL-based data warehouse solutions[13]

The writers conclude that the future of NoSQL-based DW is exciting, based on their experiences while designing and using prototypes:

1. NoSQL-built DW will put the advantages of classic DW technologies into line with the accessibility of web times like Google-style search.
2. NoSQL-based DW has the possibility that traditional DW frameworks will not be able to provide new data processing solutions.
3. The one thing that allows flexible NSQL-based DW development plans is a transparent de-normalization process that unifies design and production aspects.
4. To modify the creation of NoSQL based DW use, NoSQL reporting tools for NoSQL data storage are needed.

5.11 A framework for the cryptographic verification of java-like programs[14]

They used an automatic tool, called Joana, to test the failure of a Java software to prove the concept of a system. This study indicates that the software does not have logical differentiation. In this paper, a general framework for the determination of Java (-like) programs with system analysis tools were introduced which could verify (standard) non- interference characteristics of Java systems, but which a priori could not cope with cryptographic and cryptographic adversaries, i.e. probabilistic polynomial-time opponents. An idea is a new approach that incorporates program interpretation and safety-based modelling techniques. The Java-like language Jinja+ a contains a precious fragment of Java is defined and demonstrated in the framework.

5.12 IDCrypt: A multi-user searchable symmetric encryption scheme for cloud applications[15]

In this article, a comprehensive and quantitative analysis through index-based SE and token-based SE schemes was performed. We defined the IDCrypt design SSE schemes and evaluated their security. To satisfy encrypted quest, IDCrypt constructs web indexes at proxies with encrypted data identifiers. They also developed a token-adjustment search scheme with various indexes. To exchange encrypted data among different users, the TLES two-layer encryption scheme was proposed to transfer secret keys between different proxies. However, the experimental results demonstrate that the primary sharing system currently produces relatively low overhead.

5.13 A note on certificate-based encryption[7]

This paper suggests two certificate-based IBE systems to escape the main escrow problem. For building, asymmetric bilinear pairings are used and identities converted to an integer. The first method is Gentry’s system extended. Together with Gentry’s method, the encryption algorithm eliminates one combination. The method is an element IBE scheme. The scheme will achieve fine-grained access control.

5.14 A network coding and DES based dynamic encryption scheme for moving target defense[16]

The simulation results show that the running ratio of the proposed scheme is relatively less than or akin to the triple DES. The NC nature of the proposed scheme makes it endow the dynamic, active and random characteristics within the concept of Moving Target Defense (MTD). These were achieved by combining both the DES and the network-coding characteristic. It consists of 4 steps:

5. Inner Layer Encryption Embedding NC.
6. Middle Layer DES Encryption.
7. Outer Layer Encryption Embedding NC.
8. Dynamic Update of the Ciphertext.

5.15 Asymmetric physical layer encryption for wireless communications[17]

In this paper, a cryptographic basic for an asymmetric physical layer encryption system was developed, providing a new direction for PLE system design. An asymmetric encryption scheme based on elliptic curve cryptography is implemented. The scheme does not need to execute first delivery on a private channel compared to the existing. Symmetric PLE system and is more suited for multi-user communication scenario. The physical layer signal can be secured, and protection improved relative to the upper layer public-key encryption scheme. Research and simulation show that the suggested algorithm has higher constellation instability and the same BER performance as an unencrypted process. The approach has low latency and versatility and is easy to implement throughout hardware for specific 5 G applications.

6 Proposed Methodology

1. JDBC - is designed to communicate with different database management frameworks within a Java application.
2. Java EE - a compilation of Java language standards and related documents detailing the application platform design for medium and large enterprises.
3. Hibernate - it is an object-relational Java programming language mapping tool. This provides a framework for a relational database to map an object-oriented domain model.
4. SQL - The declarative programming language for the creation, modification and administration of data in a relational database that is handled by the appropriate database system.
5. Threads - allows you to code in such a way as to allow several tasks to run simultaneously in the same program.
6. Tomcat - enables you to run web applications, and includes several self-configuration programs.
7. IntellJ idea - is an Integrated Software Development Environment (IDE) written in Java.

These technologies have been used in the following way:

1. JDBC – grants connection between Java application and database.
2. Java EE – used as the basis for the project, and also allows you to use additional features that are present only in Enterprise Edition. Maven – is part of Java EE and allows you to connect additional libraries without unnecessary problems.
3. Hibernate – the same as JDBC, but works faster and safer compared to JDBC, on the other hand, has fewer functionality possibilities.
4. Threads – gives the ability to build the connection between chat clients of different users, receive and send messages.
5. SQL – designed for managing data in relational databases (such as MySQL, PostgreSQL, Oracle SQL etc.)
6. IntelliJ idea – used as a builder for application.
7. Tomcat - used as a standalone web server.

The application is written in Java using Maven to group assemblies that are necessary for maintaining stable work of application, such as Hibernate and JDBC. The application has such functionality as - Login, registration, real-time messaging. Instead of raw SQL language, it is used by JDBC and Hibernate - it is much safer. Moreover, the project is already ready for installation on the server. All you need is just upload it to the server, and you can use it. Since working with Hibernate is much safer in contrast to JDBC. Accordingly, it is used to work with account management functions such as Login and registration. Whereas JDBC, unlike Hibernate, supports the ability to dynamically, generate tables, which is why it has excelled as a kind of driver for creating a contact list for each of the registered users. Where everything is will be saved in MySQL database. For messaging, the app uses Threads, as they are a kind of representative of the user in the network. Their work is carried out as follows: when the user is connected (A) to the channel in which it is or will be the exchange of messages, a new thread is generated which then ties itself to the user. Moreover, later, this same thread also transfers the message to the server from where the thread of another user (B) picks it up and displays it on the user’s screen (B) then goes into standby mode, waiting for whether it will carry the message to the server or again pick it up and display it on the screen.

During development, it was found that Hibernate is a better choice for working with databases than JDBC because of its simplicity and speed of processing requests, on the other hand, hibernate does not have the same flexible functionality as JDBC. For example, in order for each new user to have their contact book during registration, JDBC is used, at each registration, JDBC takes the username. It creates a new table in the database with the username as the access key to it. Whereas in Hibernate it is not possible to freely specify a table name, it must first be created in the database and initialized in the code, after which it can only be used in Hibernate. The existence of the Spring framework can make it much easier to work with servers since Spring[16], [18], [19] itself was designed to simplify the programmer and the machine at more sophisticated levels with the help of things like SOAP and RESTful[20], [21] it is possible to raise, install and run your website including the server in this list. Which, in comparison with pure code writing, provides advantageous advantages, such as easier writing and an already built-in code optimizer that will help achieve maximum program-machine collaboration. However, because it is not possible to test the results of XML code execution because when executing a section of code associated with the Spring Framework, the main execution threads in IntelliJ idea go to an XML file where it is not possible to follow them.

MySQL database management Manager is ideally suited for this project because it is the property of Oracle and is not inferior to their main application Oracle DB and is also convenient and half free for use for commercial and personal purposes in contrast to its closest competitors from Microsoft etc. The application was also tested on the Tomcat server, and as a result, it was found that the application is well compatible with this type of server. On the other hand, using Sockets directly has its significant drawback – it is tough to predict where an error may occur, and if this happens, then it will not be possible to find out where it appeared using conventional methods. In contrast, the Spring framework offers a higher level of error tolerance and reliability than when using sockets directly. However, this method also has its drawbacks, such as complexity in management, the same inability to track code execution and auto-linking of components using the @autowired annotation. Sometimes there are failures, and this annotation can link the wrong components, which will inevitably lead to application failures and errors. Java EE (Enterprise Edition) Java created for commercial development, also known as J2EE is a set of modifications that simplify the management of application development including Maven-designed, to simplify the loading of various add-ons. For example, Hibernate, JDBC, Spring etc. Moreover, Junit – thanks to Which you can test the app in different circumstances without running the app itself every time. JUnit uses the annotation system as well as Spring, which makes both quite similar and simplifies their use. Java EE itself is no different from Java SE except for the same modifications of which there is a massive number in Java EE and on the other hand, which are almost impossible to run on Java SE.

7 Results and Discussion

7.1 Login

The user opens the program; if he does not have an account, then he must go through a registration, then logs into the system under his account and begins his work in the system by adding his friends to the contact list (all these operations are done using MySQL, Hibernate and JDBC), after that when choosing another user from your contact list, you can start messaging, which happens using threads and synchronized methods. A more detailed explanation is provided below: The Hints Texts class displays hints on empty fields where you need to enter data when you click on it. It immediately disappears and allows you to enter your text. After entering your data and clicking on the "Log In" button, the User Getters class is called where the data of the "Phone Number" and "Password" fields are transferred and compared with the database. If it returns "true", then the data is selected using the Hibernate class and hibernate config.xml after which all the data that were found is written to the Users class and transferred back to the Login class. The chat client class is called, and data from phone Number and port fields is passed there. It also starts the server with the transferred data from the variable's "name" and "port" (you can do it without "name" if you do not want to add user names to the messages area). After pressing the button "Registration", it calls Registration class.

7.2 Registration

After calling the "Registration" class, the registration form will be called (4.3). It also uses the "Hints Texts" class to display tooltips on text fields. By clicking on the "Finish" button, a random 4-digit number is generated which is passed to the "User Getters" class and checked against the table for the presence of this number in the database. If it returns null, it starts the transfer of data entered into text fields.

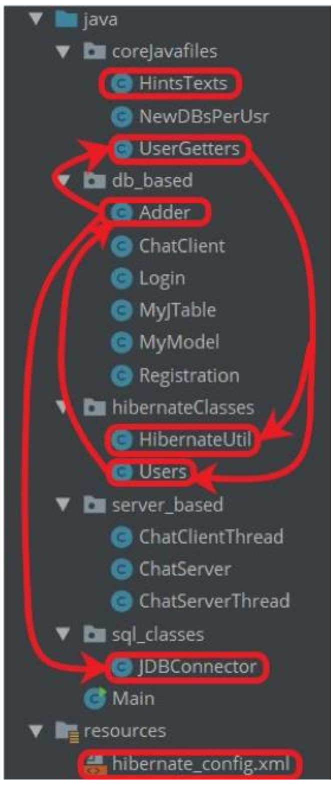


Fig. 1. Scheme of the working cycle

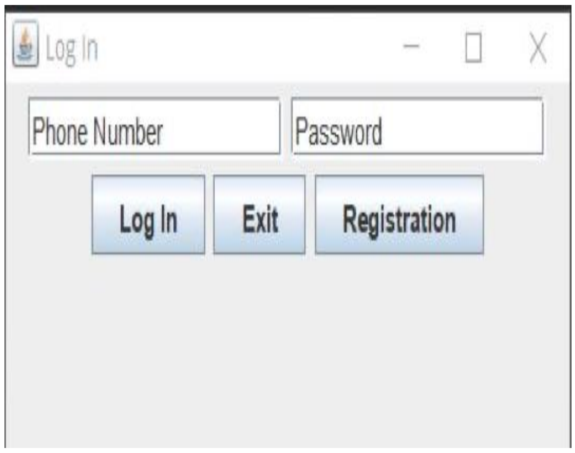


Fig. 2. Login

The class where they are stored as a temporary array, and then this data array is transferred to the “User Getters” class from where they are sent using the Hibernate to the database and stored in the Users table. After that, the phone number entered by the user is sent to the "NewDBsPerUsr" class and creates a new table in the database using JDBC with the user’s phone number instead of the table name, which will be used as users list of contacts in the future. Then it calls the “Login” class and removes this form. The full cycle of work is shown in figure 1.

7.3 Chat client

After the form is generated, data is loaded from the user’s contact list table (4.5). AS it is shown in figure 6. It uses the class “MyJTable” which takes the class “MyModel” and inherits some of its parameters. The class “MyModel" is used to display information from the database on the screen (table “MyJTable”). Additionally, “MyModel” allows you to modify the output of data on the screen, for example, generating an empty line after all entries for the subsequent addition of new ones. When a user is selected from the contact list, the classes “ChatServer”, “ChatClientThread” and “ChatServer-Thread” are launched. First, the column and row are selected using JDBC, Id, and the fourth column with the named port, which is in the database, are taken. After that, the port is saved into a Variable and recorded together with the server address in the "Socket" class, then "Socket" is sent to "ChatClientThread" and the class of the stream "DataOutputStream" which is then picked up by the stream class “DataInputStream” in the class "ChatClientThread". Moreover, the data recorded to the “Socket” class are sent to the “ChatServer” class through the “ServerSocket” class. This action creates a connection between the server and the client. In the "ChatServer" class, a call is made to the "ChatServerThread" class, which generates a thread and saves it into an array of threads in the “ChatServer” class (4.7). Also, it waits until the next call of this thread. That would happen after the send button is clicked.

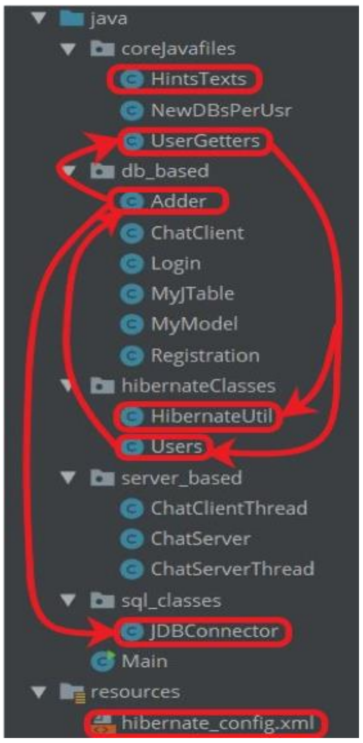


Fig. 3. Scheme of the working cycle

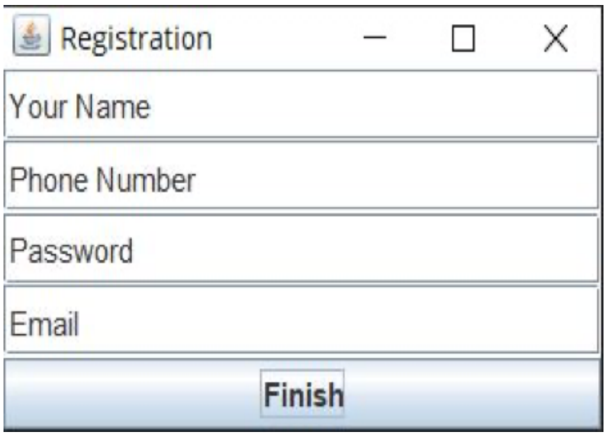


Fig. 4. Registration Form.

7.4 Insert

Uses the “Hints Texts” class to display tooltips on text fields. After clicking on the "Add" button in the "ChatClient" form, a form consisting of two buttons, a table and a text field is generated (figure8). In-text field, the user enters the phone number of the person he wants to find, after clicking on the "Search" button it sends a request to the user database and checks against the entered number (figure 4). The user that was found is returned to the array and displayed on the table by using Hibernate (figure.3).

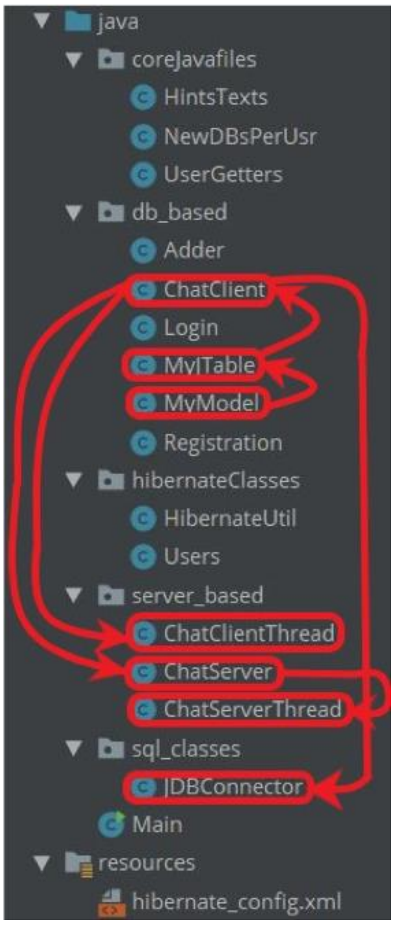
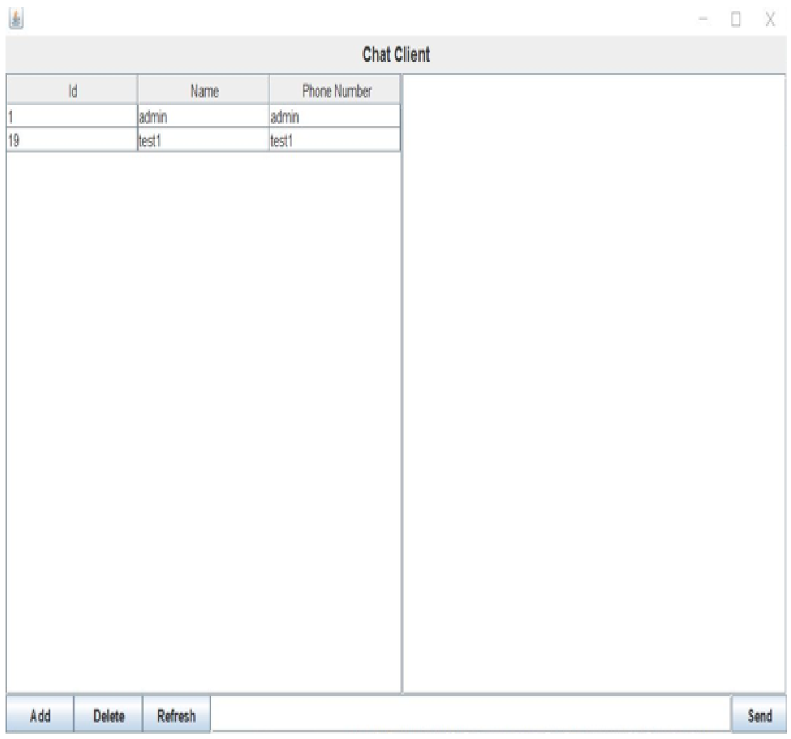
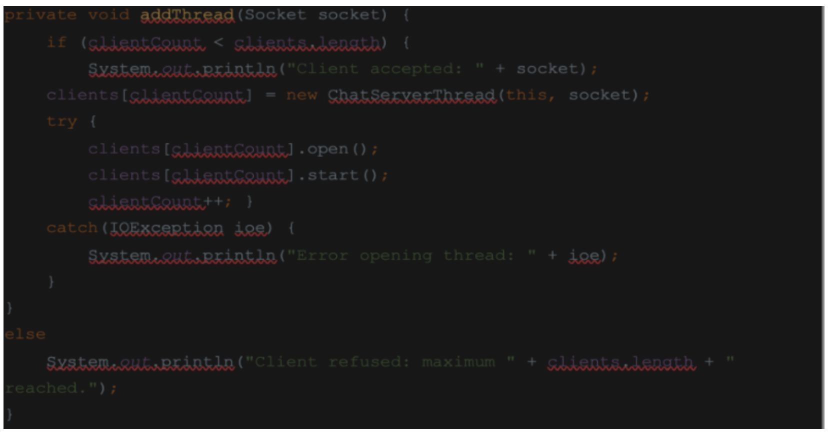


Fig. 5. Scheme of the working cycle



Id	Name	Phone Number
1	admin	admin
19	test1	test1



```
private void addThread(Socket socket) {
    if (clientCount < clients.length) {
        System.out.println("Client accepted: " + socket);
    clients[clientCount] = new ChatServerThread(this, socket);
    try {
        clients[clientCount].open();
        clients[clientCount].start();
        clientCount++; }
    catch(IOException ioe) {
        System.out.println("Error opening thread: " + ioe);
    }
}
else
    System.out.println("Client refused: maximum " + clients.length + "
reached.");
}
```

Fig. 7. Code to create a new thread.

If there is such a user and the record was displayed, then the “Add” button is activated, when clicked, the record from the table is transferred to the user database (also known as the contact list) by using the “JDBConnector” class (Figure.5).

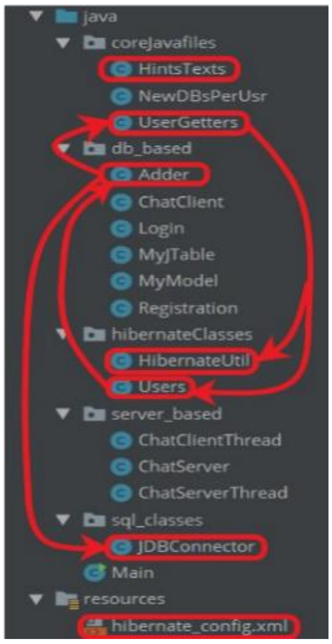


Fig. 6. Scheme of the working cycle

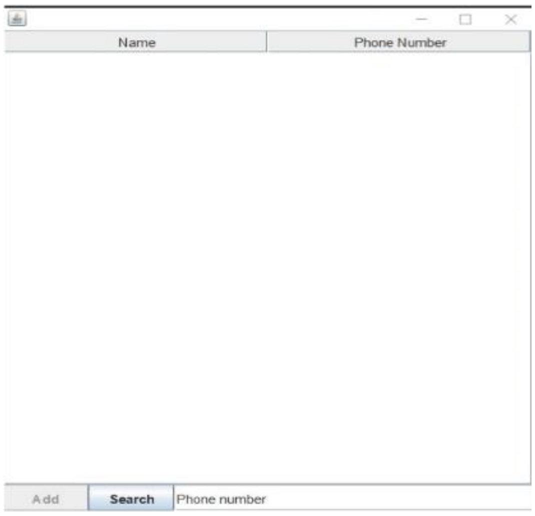
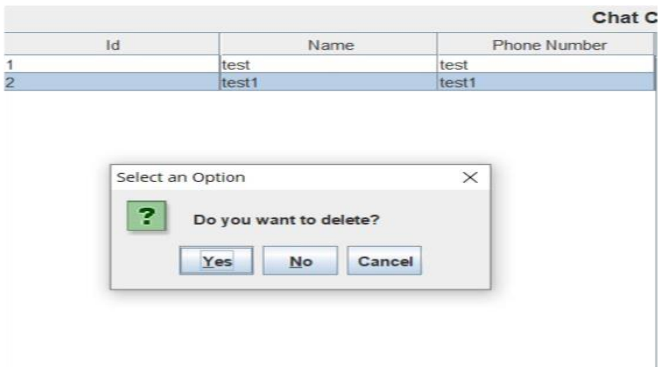


Fig. 7. Search for a new contact

7.5 Delete

After clicking on the “Delete” button, a window pops up on the screen asking to confirm the deletion of the record (4.11). If the "Yes” button is clicked, the remove method is called in the "JDBConnector" class to which the parameters of the first "Id" column are transferred and the SQL query "Delete from where " is executed (4.12). Then the "refresh" method is executed to update the table. If you click on the "No" button. As a result, the question window simply closes.



Id	Name	Phone Number
1	test	test
2	test1	test1

Fig. 12. Remove contacts.

7.6 Chat between users

When a user is selected from the contact list, a connection is made to the server and user port, which is selected using JDBC fetch from the database. If the port is not found in, the list of running server ports, the message "User is offline!" will be displayed on the screen as shown in the image above.

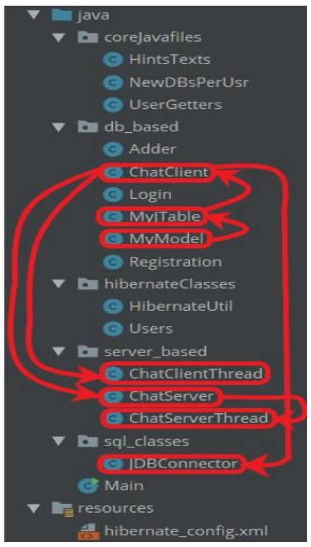
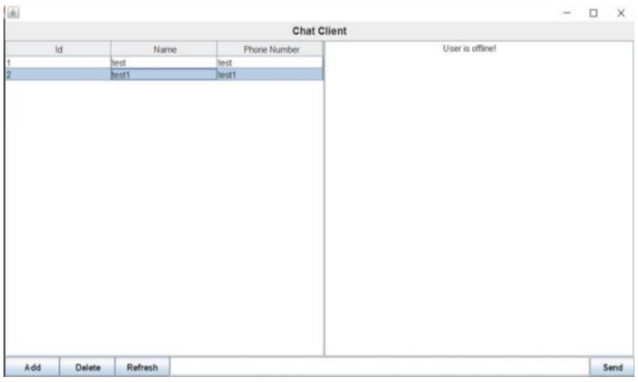


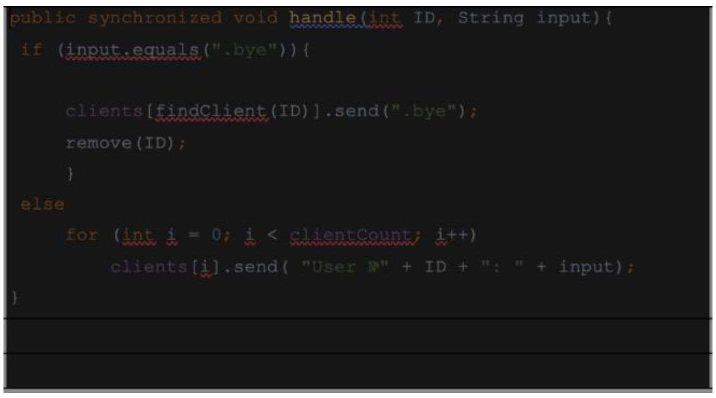
Fig. 8. Scheme of the working cycle



Id	Name	Phone Number
1	test	test
2	test1	test1

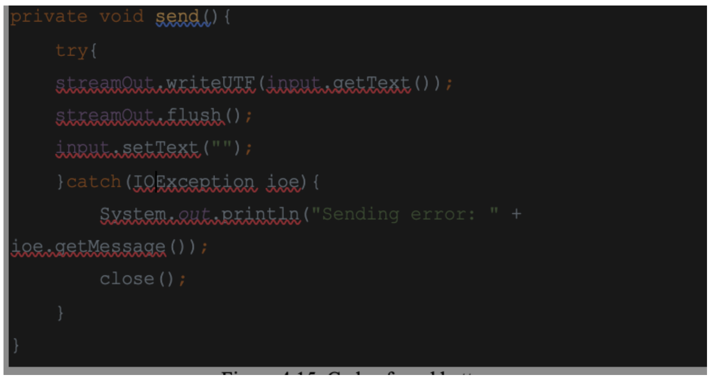
User is offline!

Fig. 9. Check if the user is online



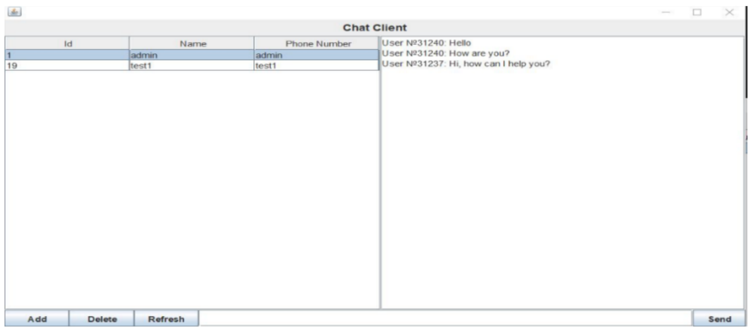
```
public synchronized void handle(int ID, String input){
 if (input.equals(".bye")){

     clients[findClient(ID)].send(".bye");
     remove(ID);
     }
 else
     for (int i = 0; i < clientCount; i++)
         clients[i].send( "User №" + ID + ": " + input);
}
```

Fig. 15. Code that shows username in chat.



```
private void send(){
    try{
    streamOut.writeUTF(input.getText());
    streamOut.flush();
    input.setText("");
    }catch(IOException ioe){
        System.out.println("Sending error: " +
ioe.getMessage());
        close();
    }
}
```

Fig. 16. Code of the send button.



Id	Name	Phone Number
1	admin	admin
19	test1	test1

User №31240: Hello
User №31240: How are you?
User №31237: Hi, how can I help you?

Fig. 17. Chat in the middle of work.

After connecting to the port of the user with whom there is a desire to start exchanging messages, entering a message and clicking on the "Send" button, the "send" method is called (4.15). The message is picked up by the stream and transferred to the screen of another user using the "handle" method (4.16). In which the loop finds the thread ID in the stream array and sends the result to the "send" method in the "ChatServerThread" class, which in turn uses the "DataOut- putStream" class to transfer the data to the "ChatClientTread" class where it will be caught by the "DataInputStream" class (4.14) and displayed on the screen. As can be seen in Figure 4.17, messages are successfully displayed on both users. In this example, User No. is used instead of the username to show the IDs of the threads that are delivering messages between clients. In the future, these IDs can be replaced with user names or even removed and substituted with a different user identification system.

8 Conclusion

In this article, an application was written with the most optimal in comparison to the functionality and speed of execution. Also, the Hash encryption method was applied[11] for greater security of messaging over the Internet, but this application is designed for PCs and is inferior in optimization to its closest relatives written for Android[5], [18], [19], [22] or for browsers written in JavaScript [23]–[25]. However, if it is possible to achieve a harmonious combination of technologies and methods specified in the 2nd Chapter. Then it will be able to achieve new security peaks against message hacking[16], [19], [26], [27] and SQL injections [28], [29]. But, since this implementation will lead to a deterioration in the optimization of the application and an extra load on the server, then it will be needed methods that can stabilize and improve the connection of the application with the database[12], [30] and the server [31]–[33]. As a result, we can say that in comparison with applications for Android or the browser, the PC version is inferior to them in the speed of execution. However, on the other hand, the PC version also has great potential in development. In practice, this application is best suited for use within the company since the company has a limited number of employees and it will be easier to configure the application, plus the added protection of the company will be an additional level of security. However, if used for a global application, then you will need to improve the security level and change the database to a dynamic one[4].

9 References

[1] R. Klemm, "Practical guidelines for boosting Java server performance," 1999, pp. 25–34.
[2] R. Klemm, "Practical guidelines for boosting Java server performance. Proceedings of the ACM 1999 conference on Java Grande - JAVA '99." https://doi.org/10.1145/304065.304090
[3] H. Abbes and E. Al, "Big Data Integration: A MongoDB Database and Modular Ontologies based Approach," Procedia Computer Science, vol. 96, pp. 446–455, 2005. https://doi.org/10.1016/j.procs.2016.08.099
[4] L. Rocha, F. Vale, D. Barbosa, and F. Mourão, "A Framework for Migrating Relational Datasets to NoSQL *."
[5] P. Mehrotra, T. Pradhan, and P. Jain, "Instant Messaging Service on Android Smartphones and Personal Computers," 2014.
[6] X. Xie, C. Ren, Y. Fu, J. Xu, and J. Guo, "SQL Injection Detection for Web Applications Based on Elastic-Pooling CNN," IEEE Access, vol. 7, pp. 151475–151481, 2019. https://doi.org/10.1109/access.2019.2947527
[7] Y. Qi, C. Tang, M. Xu, and B. Guo, "A note on certificate-based encryption," in IET Conference Publications, 2014, vol. 2014, no. CP653.
[8] C. Fu and B. G. Ryder, "Exception-chain analysis: Revealing exception handling architecture in Java server applications," in Proceedings - International Conference on Software Engineering, 2007, pp. 230–239. https://doi.org/10.1109/icse.2007.35
[9] D. Shuo, "Request Dispatching Algorithms for Web Server Ousters Based on Load Balancing," TSINGHUA SCIENCE AND TECHNOLOGY, vol. 4, no. 4, pp. 1620–1623, 1999.
[10] A. De Caro and V. Iovino, "jPBC: Java pairing based cryptography," in 2011 IEEE symposium on computers and communications (ISCC), 2011, pp. 850–855. https://doi.org/10.1109/iscc.2011.5983948
[11] N. Y. Goshwe, "Data Encryption and Decryption Using RSA Algorithm in a Network Environment," 2013.
[12] C. Gould, Z. Su, and P. Devanbu, "JDBC checker: A static analysis tool for SQL/JDBC applications," in Proceedings - International Conference on Software Engineering, 2004, vol. 26, pp. 697–698. https://doi.org/10.1109/icse.2004.1317494
[13] Z. Bicevska and I. Oditis, "Towards NoSQL-based data warehouse solutions," downloadpaper.com, 2017. https://doi.org/10.1016/j.procs.2017.01.080
[14] R. Küsters, T. Truderung, and J. Graf, "A framework for the cryptographic verification of Java-like programs," in 2012 IEEE 25th Computer Security Foundations Symposium, 2012, pp. 198–212. https://doi.org/10.1109/csf.2012.9
[15] G. Wang, C. Liu, Y. Dong, P. Han, H. Pan, and B. Fang, "IDCrypt: A Multi-User Searchable Symmetric Encryption Scheme for Cloud Applications," IEEE Access, vol. 6, pp. 2908–2921, Dec. 2017. https://doi.org/10.1109/access.2017.2786026
[16] H. Tang, Q. T. Sun, X. Yang, and K. Long, "A Network Coding and des Based Dynamic Encryption Scheme for Moving Target Defense," IEEE Access, vol. 6, pp. 26059–26068, May 2018. https://doi.org/10.1109/access.2018.2832854

[17] W. Li, D. McLernon, K. K. Wong, S. Wang, J. Lei, and S. A. R. Zaidi, "Asymmetric Physical Layer Encryption for Wireless Communications," IEEE Access, vol. 7, pp. 46959–46967, 2019. https://doi.org/10.1109/access.2019.2909298

[18] J. Lee, S. J. Lee, and P. F. Wang, "A Framework for Composing SOAP, Non-SOAP and Non-Web Services," IEEE Transactions on Services Computing, vol. 8, no. 2, pp. 240–250, Mar. 2015. https://doi.org/10.1109/tsc.2014.2310213

[19] A. Mahajan, M. S. Dahiya, and H. P. Sanghvi, "Forensic Analysis of Instant Messenger Applications on Android Devices," International Journal of Computer Applications, vol. 68, no. 8, pp. 38–44, Apr. 2013. https://doi.org/10.5120/11602-6965

[20] J. M. Tekli, E. Damiani, R. Chbeir, and G. Gianini, "SOAP processing performance and enhancement," IEEE Transactions on Services Computing, vol. 5, no. 3, pp. 387–403, 2012. https://doi.org/10.1109/tsc.2011.11

[21] P. A. Castillo, J. L. Bernier, M. G. Arenas, J. J. Merelo, and P. Garcia-Sanchez, "SOAP vs REST: Comparing a master-slave GA implementation," May 2011.

[22] K. D. Sowjanya, "Instant Message Transfer between Two Smart Phones Using Wi-Fi," International Journal of Advanced Engineering, Management and Science, vol. 2, no. 12, pp. 1949–1951.

[23] S. Tilkov and S. Vinoski, "Node.js: Using JavaScript to build high-performance network programs," IEEE Internet Computing, vol. 14, no. 6, pp. 80–83, Nov. 2010. https://doi.org/10.1109/mic.2010.145

[24] F. Shahzad, "Modern and Responsive Mobile-enabled Web Applications," in Procedia Computer Science, 2017, vol. 110, pp. 410–415. https://doi.org/10.1016/j.procs.2017.06.105

[25] K. Shuang and K. Feng, "Research on Server Push Methods in Web Browser based Instant Messaging Applications," 2013. https://doi.org/10.4304/jsw.8.10.2644-2651

[26] G. M. Waleed, G. M. Al-Saadoon, M. Perlis, K. Pusat Pengajian Seberang Ramai No, J. Satu, and T. Seberang Jaya Fasa, "A Platform to Develop a Secure Instant Messaging Using Jabber Protocol Enhancing security level for summative assessment View project Higher Education Academic View Project A Platform to Develop a Secure Instant Messaging Using Jabber Protocol," Article in Journal of Computer Science, vol. 5, no. 9, pp. 661–665, 2009. https://doi.org/10.3844/jcssp.2009.661.665

[27] A. Deb and S. Sinha, "Bluetooth Messenger: An Android Messenger app based on Bluetooth Connectivity," vol. 3, pp. 61–66. https://doi.org/10.9790/0661-16336166

[28] Q. Li, W. Li, J. Wang, and M. Cheng, "A SQL Injection Detection Method Based on Adaptive Deep Forest," IEEE Access, vol. 7, pp. 145385–145394, 2019. https://doi.org/10.1109/access.2019.2944951

[29] S. W. Boyd and A. D. Keromytis, "SQLrand: Preventing SQL injection attacks," Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 3089, pp. 292–302, 2004. https://doi.org/10.1007/978-3-540-24852-1_21

[30] R. Lawrence and T. Mason, "Dynamic Database Integration in a JDBC Driver. DYNAMIC DATABASE INTEGRATION IN A JDBC DRIVER," 2005. https://doi.org/10.5220/0002552103260333

[31] B. Yulianto and E. al, "Architecture and Implementation of Instant Messaging in Educational Institution," Procedia Computer Science, vol. 59, pp. 5–13. https://doi.org/10.1016/j.procs.2015.07.331

[32] M. Vieira and N. Laranjeiro, "Comparing web services performance and recovery in the presence of faults," in Proceedings - 2007 IEEE International Conference on Web Services, ICWS 2007, 2007, pp. 623–630. https://doi.org/10.1109/icws.2007.63

[33] A. De Caro and V. Iovino, "jPBC: Java Pairing Based Cryptography," in Proceedings - IEEE Symposium on Computers and Communications, 2011, pp. 850–855 https://doi.org/10.1109/iscc.2011.5983948

10 Authors

Dr. M. Aqel (Musbah J Aqel) is an assistant professor in department of Management Information system at Cyprus International University, TRNC. At present, he is lecturer in Department Management Information system at Cyprus International University. His main research interests are expert system, security algorithm for computer networks, and E- business software applications. His publications appear in international refereed journal. He is served Editorial Board and refereed for list of inter- national journal.

Dr. M. Sokiyna (Munsif Y Sokiyna) obtained his Bachelor's degree in Management Information System (MIS) from Jadara University. Then he obtained his Master's degree in Electronic Business (E-business) and he is now PhD candidate in Management Information System majoring in E-business Information System and E-business applications software in Cyprus International University faculty of Sciences & in- formation technology in TRNC. His main research interests Cloud computing, Big Data, Business Analytics.

Dr. O. Naqshbandi (Omar Naqshbandi) is a Lecturer in Hawler Medical University. He has been completed Bachelor's degree in Computer Engineering from Near East University- North Cyprus, and Master's degree in Computer Networks Principles and Practice, Hertfordshire University, London - Hertfordshire, UK. Currently, he is PhD candidate in Management Information System majoring in e- Government Information Systems, Cloud computing in Cyprus International University faculty of Sciences & information technology. Address: Erbil-Kurdistan Region-Iraq.

P. Valentyn (Pertsevoi Valentyn) obtained his Bachelor's degree in Software Development from Eu- European University in 2106. Then he obtained his Master's degree in Computer Information system (CIS) from Cyprus International University in 2019 faculty of Sciences & information technology in TRNC.

Article submitted 2020-04-24. Resubmitted 2020-05-26. Final acceptance 2020-05-27. Final version published as submitted by the authors.