
VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

## School of Computer Science and Engineering

VIT Chennai

Vandalur - Kelambakkam Road, Chennai - 600 127

# Project Report

## CHIT-CHAT(ANDROID CHAT APPLICATION)

**AKASH KUMAR**

**[21MCA1021]**

**NAVEEN GUPTA**

**[21MCA1042]**

**Master of Computer Application (MCA)**

(akashals213@gmail.com)

(naveen.gupta2021@vit.ac.in )

**Under the guidance of Dr. M. PREMLATHA**

**School of Computer Science and Engineering**

## ABSTRACT

Communication via the internet is turning into vital nowadays. Online verbal exchange allows the users to speak with other human beings in a fast and handy way. Considering this, the web verbal exchange utility need to be capable of share the texts or photographs or every other documents in a faster way with minimum put off or without a delay. Firebase is one of the systems which gives a real-time database and cloud services which allows the developer to make these applications effectively.

Instant messaging can be taken into consideration as a platform to maintain Communication. Android gives higher platform to increase various packages for immediate messaging compared to other systems inclusive of iOS. The primary objective of this project is to offer a software utility for the launching a real-time Communication among users. The system evolved on android will permit the users to speak with another consumer through textual content messages with the assist of the internet. The system requires for each the device to be connected through internet. This application is primarily based on Android with the backend provided by means of Google Firebase.

KEYWORDS: Android application, chat application, Firebase, Kotlin, JAVA.

## I. INTRODUCTION

In the real world Communication plays a vital role. People were communicating with each other through various applications or mediums. In the beginning people communicated with each other through the letters or other sources, as these mediums could take a lot time to supply the content.

Cell phones are some other medium of conversation however the drawback is for any limited or small message which need to be passed to any other person then a phone call is not a perfect way. The developers then seemed to implement a text-based Communication which would permit an in immediately communication carrier. In 1984, the concept of SMS was developed in the Franco-German GSM cooperation by “Friedhelm Hillerbrand” and “Bernard Ghillebaert”. The implementation of SMS become the restrained or of limited size i.e., 128 bytes, after the rise of the smartphones from a decade many messaging applications were developed. A few are Bluetooth based and a few had been internet-based along with hike, telegram, WeChat and others.

Android is an operating system for mobiles which was developed by Google.

This operating system permits the packages to be used on Mobiles. As it developed by Google, Android users can develop android applications and can be sell via android application stores such as: Play Store. Firebase is a NoSQL database that makes use of sockets which permits the users to store and retrieve the data from the database. An Android version should be greater than 2.3, android studio1.5 or higher version, and android studio projects are the prerequisites to attach the firebase to an android utility.

Firebase presents a various kind of services consist of:

**Firebase Authentication:**

Firebase Authentication is useful to both developers and the users. Developing and maintaining sign-in set-up may be a bit difficult and time taking. Firebase provides an easy API for sign in. It also provides the data backup using real time databases.

**Firebase cloud**: For storing the data such as video, text, pictures building the infrastructure would be difficult and expensive for a new developer so the firebase provides the platform of cloud storage .

**Real time database**: It is a cloud hosted NoSQL database. Apart from the authentication, cloud service and real time databases firebase also provides a service for crash reporting

**Crash Reporting**: when some unexpected crashes occur in any applications it may be difficult to conclude why the application crashed. Firebase provides crash reporting service to deal with these crashes.

**KOTLIN**

Kotlin is a cross platform , statistically typed, general purpose programming language with type interface. Kotlin is designed to interoperate fully with java, and the JVM version of Kotlin's standard library depends on the Java Class Library, but type inference allows its syntax to be more concise. Kotlin mainly targets the JVM, but also compiles to JavaScript (e.g., for frontend web applications using React or native code via LLVM (e.g., for native iOS apps sharing business logic with Android apps). Language development costs are borne by JetBrains, while the Kotlin Foundation protects the Kotlin trademark.

On 7 May 2019, Google announced that the Kotlin programming language is now its preferred language for Android app developers.[7] Since the release of Android Studio 3.0 in October 2017, Kotlin has been included as an alternative to the standard Java compiler. The Android Kotlin compiler produces Java 8 bytecode by default (which runs in any later JVM), but lets the programmer choose to target Java 9 up to 16, for optimization,[8] or allows for more features; has bidirectional record class interoperability support for JVM, introduced in Java 16, considered stable as of Kotlin 1.5.

## II. LITRETURE REVIEW

**PAPER 1**

TITLE : Android Malware Detection Using Deep Learning
Year : 2021
Author : Omar N. Elayan, Ahmad M.Mustafa
Publication: ELSEVIER
PROBLEM STATEMENT:
Traditional Android malware detection methods, such as signaturebased methods or methods monitoring battery consumption, may fail to detect recent malware.
ABSTRACT:
The Android operating system most used operating system and first rank holder in the share market due to the system's smooth handling and many other features that it provides to their users, because of these useful feature cyber criminal got attracted to its side. There are many traditional methods available for malware detection, such as signaturebased methods or methods monitoring battery consumption, but these methods may fail to detect recent malware. Therefore, they approached a new novel method for detecting the malware in android application using Gated Recurrent Unit (GRU), that's a type of Recurrent Neural Network (RNN). In this research they extract two static features, Application Programming Interface (API) calls and Permissions from Android applications. In this research they train and test their approach using CICAndMal2017 dataset. The experimental result shows that their deep learning method performs very well over several methods with accuracy of 98.2%.
SUMMARY:
As the Android operating system occupies the leading market share and its open-source nature, its security is increasingly challenging and threatening. As its growing increasingly day by day its much important to handle the Android malware, research on Android malware detection using deep learning has been a research hotspot in recent years, but currently lacking a detailed and comprehensive introduction to the research progress of Android malware detection using deep learning. Firstly, the paper introduces the basic knowledge of Android malware detection technology. Then it sort-outs, analyzes, and summarizes Android malware detection's latest research progress based on deep learning and summarizes the Android malware detection architecture and detection scheme based on deep learning. Then it analyzes the problems and challenges of Android malware detection using deep learning.
For doing this research they have made use of two analysis phases that are:
STATIC ANALYSIS:

This phase is basically focuses on analyzing the static information obtained without running the application, such as the executable file and source code. This analysis is based on the de-compilation technology to perform reverse analysis of the code. The advantage of static analysis is that the calculation overhead is relatively low, and the detection speed is fast, but there is a disadvantage of this is that malware using obfuscation technology cannot be accurately analyzed by the static analysis. Android applications are released on the applications market in the form of APKs. In static analysis, APK tool is used to decompile the APK files and parses the AndroidManifest.xml file to extract the permissions, package names, components, environment, and intent features. IDA Pro decompiles the APK file afterwards it parses the shared library. Android applications are developed using Java. The development environment (such as Eclipse) converts Java source code into Dalvik executable files (dex files). These files can run on Android's Dalvik virtual machine. (Dex is a file format containing compiled code programmed for Android that can be interpreted by the Dalvik virtual machine but can't be read. Smali provides readable code in the smali language for converting the dex file into readable format. (Smali code is work as the intermediate code for interpretation between Java and Dalvik virtual machines.) . Smali code can be used for feature extraction it also can be used for access Java source code files to extract features such as API calls. Static analysis is decompressing the APK file to obtain the classes.
DYNAMIC ANALYSIS: Dynamic analysis is used to run an application in a sandbox environment or on a real device. Its work is to monitor the application status when the application is running and collect the information about the application's behavior and analyze a series of data information (log, network traffic). Application behavior is basically monitored by accessing the private data or using the restricted API calls. By the use of Dynamic analysis we can identify the malicious behaviors that we can't detect by the static analysis methods. The advantage of dynamic analysis is that it can deal with malware using obfuscation techniques, but there is an disadvantages of this is that it take long time for analysis and detection , and also the calculation cost is high. The dynamic analysis method collects behavior information when the Android application is running and transforms it into features.

## PAPER - 2

TITLE: SAMADroid: A Novel 3-Level Hybrid Malware Detection Model for Android Operating System
Year : 2018
Author: S. Arshad, M. A. Shah, A. Wahid, A. Mehmood, H. Song and H. Yu
Publication: IEEE ACCESS

PROBLEM STATEMENT:

The existing research clearly lags in detecting malware efficiently and accurately.

ABSTRACT:

Android is the most extensively used operating system, and its quickly growing property has attracted the malware developers. Android allows you to download and install apps from unofficial app stores. This allows malware developers to distribute repackaged harmful apps through third-party app stores and attack Android devices. To keep Android devices safe from malware, a vast variety of malware analysis and detection solutions have been created, which uses static analysis, dynamic analysis, or hybrid analysis. However, current researches are fail in terms of detecting malware quickly and accurately. Multilayer analysis is essential for accurate malware detection, which demands a significant amount of hardware resources on resource constrained mobile devices. SAMADroid, a novel 3-level hybrid malware detection model for Android operating systems, is proposed as an efficient and accurate solution to this challenge in this study. There are several aspects to the research contribution.

SUMMARY:

In this research work they have focused on the development of a malware detection system that can detect the malwares on the Android devices while ensuring the low resource consumption. In this, they thoroughly investigated many of the existing malware detection and prevention techniques. Based on the benefits and limitations of existing antimalware techniques, they formulated the problem that existing research lags in detection of Android malwares accurately while ensuring the low consumption of hardware resources of Android devices. Thus, they proposed 3-level hybrid malware detection model for Android operating system. These levels are--

LEVEL 1: STATIC AND DYNAMIC ANALYSIS:

At this level, the hybrid of static and dynamic analysis provides a highly accurate analysis as it combines the benefits of two analysis techniques.

Through static analysis, it scans all the code of application and analyzes the malicious behavior of application without executing it.

In dynamic analysis phase, system executes the application on the real device and analyzes its runtime behavior which also includes the monitoring of dynamically loaded and decrypted code. System calls are used as dynamic features. Applications installed on real Android devices are analyzed by system call tracing. These system calls allow us to overcome the limitations of static analysis and analyze the application's behavior in real time environment.

LEVEL 2: LOCAL AND REMOTE HOST

Level 2 is a hybrid of local and remote host. Detailed static analysis is performed on the remote host to achieve highly accurate results. On the local host, dynamic analysis is performed to take the realistic inputs from the user instead of using any programmed tool, Monkey Runner, which generates non-realistic random input events. On the basis of user inputs, system call logs are generated and forwarded to the remote server. Remote server keeps on analyzing the behavior of application on the basis of logs and extracted static features.

LEVEL 3: MACHINE LEARNING INTELLIGENCE

At Level 3, the feature vectors built from analyzed features are given as input to the machine learning intelligence unit to perform the detection of malicious behavior of unknown apps and to correctly classify them. All the applications are classified as malicious or benign. In SAMADroid, the detection operation is performed at remote host, thus keeping all the training dataset in memory of server, which is a resource rich system. This ultimately reduces the memory overhead.

Till the time of research there does not exist any 3-level hybrid malware detection system. Thus, SAMADroid is a novel malware detection model which combines the benefits of static analysis, dynamic analysis and machine learning Intelligence. Also, it operates both on local host and remote host to achieve the resource efficiency. SAMADroid client application is basically designed for Android devices. It performs dynamic analysis on the device and communicates with the server for static analysis and detection results. Remote server performs the static analysis and machine learning based detection. In their experiment, they have shown that SAMADroid performs better than Drebin for static analysis. It's also observed that SAMADroid causes very negligible performance overhead on the Android device.

## PAPER 3

TITLE: Android Malware Detection Based on Factorization Machine

Year : 2019

Author : C. Li, K. Mills, D. Niu, R. Zhu, H. Zhang and H. Kinawi

Publication: IEEE ACCESS

PROBLEM STATEMENT:

Traditional methods like signature-based routines are unable to protect users from the ever-increasing sophistication and rapid behavior changes in new types of Android malware.

ABSTRACT:

As Android becomes the most popular operating system for Smartphone users in recent years, malware developers have created malicious applications for it. Because of the risk of data theft that most mobile phone users faces. Detecting malware in Android has become an important part of cybercrime detection. Traditional solutions, such as signature-based procedures, are insufficient to protect users against new varieties of Android malware's rising sophistication and rapid behavior changes. As a result, machine learning models and algorithms have been used to detect malicious activity in mobile phones for malware detection. They present a novel and highly reliable classifier for Android Malware detection based on Factorization Machine architecture and the extraction of Android app features from manifest files and source code in this study. The outcome shows that an app's numerical feature representation is accurate.

SUMMARY:

The popularity of the android Smartphone is increased, so it has many numbers of malicious applications. It's an important aspect to detect the malware on the android device for the cyber security field. In this paper, they consider the problem as considering interaction terms across features for the discovery of malicious behavior patterns in Android applications.

They evaluated their model on two Android malware datasets: DREBIN and AMD, which contain 5,560 and 24,553 samples. In order to gauge performance, for this they use the matrices such as- accuracy, false-positive rate (FPR), precision, recall, F1 and last but not least, training time. In addition, they also evaluated the performance while identifying specific families of malware, which is especially important given the growing share of banking Trojans on the internet. For this task, they focused primarily on accuracy and false-positive metrics. The features used to represent an apk file consist of app components, hardware features, permissions and intent filters from the manifest file, as well as restricted APIs, suspicious APIs and used permissions from source code. Based on the extracted features, a highly sparse vector representation was constructed for each application using one-hot encoding. After that, they proposed the use of a Factorization Machine-based malware detection system to handle the high sparsity of vector representation and model interaction terms at the same time. As it is the first technique for malware detection using FM models.

A comprehensive experimental study on two real sample malware collections, the DREBIN and AMD datasets, alongside clean applications collected from online app stores were performed to show the effectiveness of the system on malware detection and malware family identification tasks. Promising experimental results with accuracy, precision, recall and F1 scores of around or above 99% demonstrated that their method matches the performance of commercial antivirus engines and holds steady against the incredible results produced by Multi-Layer Perceptions with the benefit of taking up to 50 times less time to train.

## III. IMPLMENTATION

In this application user can communicate with his friend by typing the message also can send the emojis. To use the application user has to first login with the registered email id and password, If the user is not registered then firstly he has to register himself by using the signup form where he has to enter their email id, username, password and mobile number after pressing on the signup button user will be registered to directly the home screen will be displayed to the user.

After registering in this application it first creates a room in which all the registered user's name will be shown ,(a user can chat with any of the registered user by pressing on their username) if the logged in user wants to chat then first he select the username of the user with whom he want to chat, after that he can type the message and press the send button (a user can also send the emoji with the text message ). After pressing the send button the message will be send to the specific user and only he can see the

message. On other hand if the specific user is logged in then if he press the any users username them he will get the list of message he sends of received while chatting with that user and if the user open the chat of the user which sends the message then he will get the message and he can reply in the same manner the first user sends the message.

**Algorithm used**:

There are certain algorithm which are been used to develop the application which includes.

AUTHENTICATION:

Most of the application requires the identity of the user which will help making the data of the user safer and more secured in a cloud. Firebase provides backend, SDK and ready to use libraries which help the developer to provide authentications effortlessly.

declare and initialize Sign in variable of type private integer to function OnactivityResult
if request code equals to sign in then
display sign in successful
display chat messages
else
display sign in unsuccessful
end task

The algorithm lets the user to login into the application with a valid email id. The algorithm first initializes the variable sign in to 1. That means true. The user then enters the email id which is stored in another variable internally in the database. The email id is then verified and the result is stored in a variable request_code.

If the value of the request_code matches with the value of the variable if both the values are same then it is considered as the email is valid and the user sing in to the application. If the values do not match then the sign in will not be done and the task ends. Send and receive messages: After a successful sign in the user can now able to send and receive the messages.

Create function on click
Initialize variable of type edittext to input id
Firebasedatabase.getinstance().getrefrence().push().setValue
(new classname(input.getText.tostring(),
firebaseAuth.getinstance ().getcurrentuser.getEmail()));
Set the input to NULL
If(current username from database instance is not equal to null then)
Display welcome and username
Call function display
End function

The function on_click is a function defined and the variable of type EditText is declared and initialized to the id of input text which is retrieved from the layout xml file. Email id and username of the sender is received from the firebase database instance along with the text which need to be sent. These two are converted to string and stored in the database reference of the database root node.
When this process is done then the input is set to null and the user is allowed to send another message. Secondly, for the existing users if the current user name from the firebase database is not equal to null then the user will get a welcome screen with the previous messages and new messages by calling the message display function.

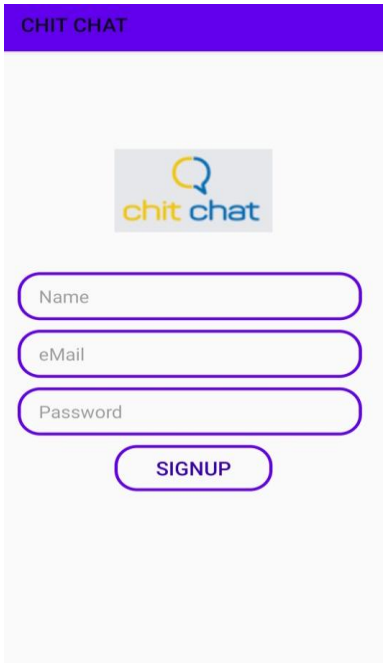
Create function display
Initialize variable list of type listView to list ID
Declare Textview variables text and user
Initialize text to text id
Initialize user to user id
Set text to model.gettext
Set user to model.getuser
Display list of messages
End function

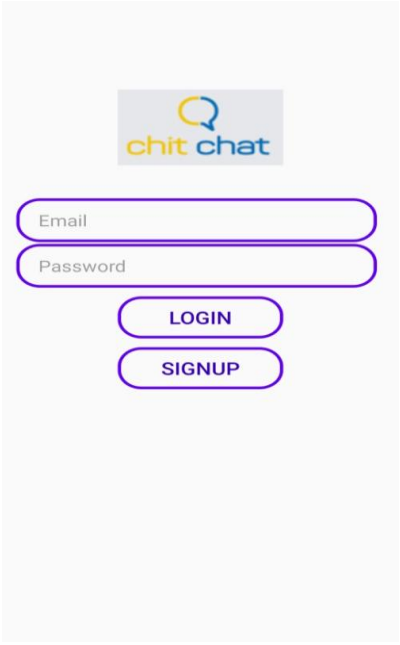
A variable of type ListView is initialized to list id. Text and user variables of type TextView are initialized to text and user id that means the sender text and name are stored in these variables. By using predefined function 'model.get' the data is stored into the variables and displayed at the time of the function call. Chat room: when the user wants to get the information about any topic then the user can search for the room by giving the keyword.

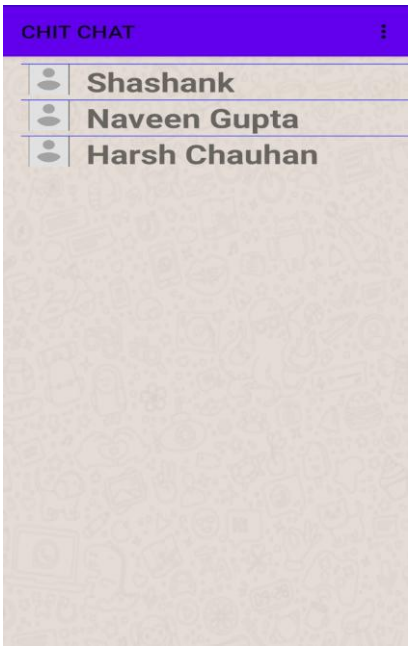
If user finds any relevant chat room then the user can directly join the room. If there is no chat room with that name existing then user can create the chat room and refer it to another users. After the users are joined in the chat room, users can decide whether the messages which they send will be displayed along with the sender name or not. If the user wants to be visible to other users then the messages are displayed along with the username.

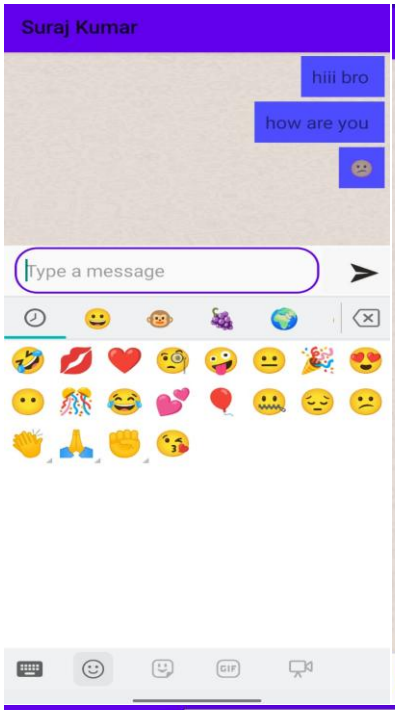
If the user does not want the actual name to be visible then a unique id is generated as a username for that particular chat room and the messages are displayed with the generated name. The chatroom can be deleted by the user who created them. The following flow chat explains the chat room work flow.

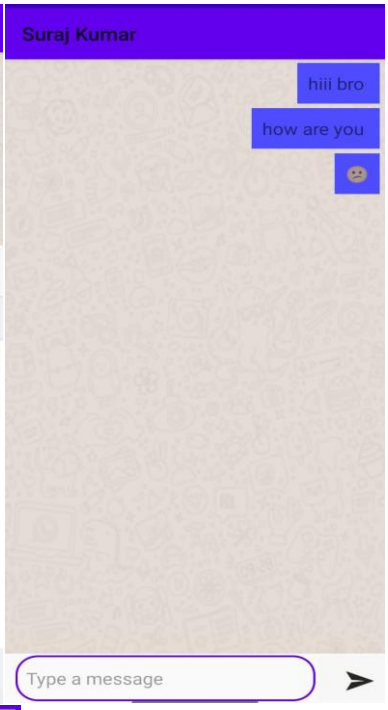
# IV. RESULTS(SAMPLES)



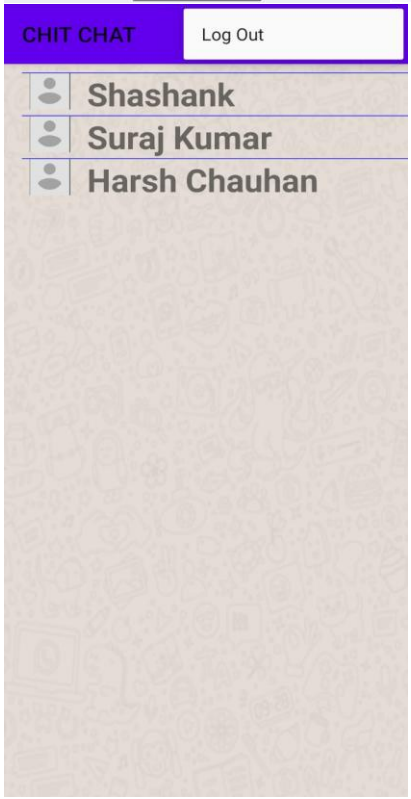




Suraj Kumar
hiii bro
how are you
Type a message


Suraj Kumar
hiii bro
how are you
Type a message


Harsh Chauhan
hiii bro
how are you
Type a message


CHIT CHAT
Log Out
Shashank
Suraj Kumar
Harsh Chauhan

## V. CONCLUSION

In a growing country it's a necessary to communicate with everyone in real time . without communication it is not possible to share our ideas or thoughts and receive others ideas and thoughts . In order to make online communication we have developed an online android chat application where users can chat with their friends also with unknown person. They can also share emoji in their chat with the text message.

## VI. REFRENCES


1. Maxmanroe. Pengertian smartphone, sistem operasi, fitur, jenis smartphone, maxmanroe.com, 2018. [Online]. Available: https://www.maxmanroe.com/vid/teknologi/mobile-app/ pengertian-smartphone.html. [Accessed: 11-Dec-2018].

2. Tasmania Police. Traffic FAQ - Tasmania Police, police.tas.gov.au, 2019. [Online]. Available: https://www.police.tas.gov.au/what-we-do/traffic-polici ng/traffic-faq/. [Accessed: 04-May-2019].

3. S. H. Ahmed, M. A. Yaqub, S. H. Bouk, and D. Kim. SmartCop : Enabling smart traffic violations ticketing in vehicular named data networks, Mobile Information Systems, vol. 16, 12 pages, May 2016. https://doi.org/10.1155/2016/1353290

4. H. Amer, R. Abd El Aziz, and M. Hamza. Investigating mobile traffic violation ticketing service in Egypt: The provider and user perspectives, International Journal of Management and Information Technology, vol. 10, no. 1, pp. 1776-1783, June 2014. https://doi.org/10.24297/ijmit.v10i1.650

5. C. Schmidt. What is Android? Here is a Complete Guide for Beginners, androidpit.com, 2016. [Online]. Available: https://www.androidpit.com/what-is-android. [Accessed: 06-Aug-2018].

6. P. Leahy. What is Java Computer Programming Language?, thoughtco.com, 2018. [Online]. Available: https://www.thoughtco.com/what-is-java-2034117. [Accessed: 06-Aug-2018].

7. M. Rouse. What is Chatting?, techtarget.com, 2005. [Online]. Available: https://searchmicroservices.techtarget.com/definition/ch atting. [Accessed: 09-Dec-2018].

8.Omar N. Elayan , Ahmad M. Mustafa, "Android Malware Detection Using Deep Learning ", Procedia Computer Science,Volume 184,2021,Pages 847-852,ISSN 1877-0509, https://doi.org/10.1016/j.procs.2021.03.106.

9. S. Arshad, M. A. Shah, A. Wahid, A. Mehmood, H. Song and H. Yu, "SAMADroid: A Novel 3-Level Hybrid Malware Detection Model for Android Operating System," in *IEEE Access*, vol. 6, pp. 4321-4339, 2018, doi: 10.1109/ACCESS.2018.2792941. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8255546&isnumber=8274985

10. C. Li, K. Mills, D. Niu, R. Zhu, H. Zhang and H. Kinawi, "Android Malware Detection Based on Factorization Machine," in *IEEE Access*, vol. 7, pp. 184008-184019, 2019, doi: 10.1109/ACCESS.2019.2958927. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8931539&isnumber=8600701