Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN : 2581 - 611X)

Volume 7- Issue 1, January 2024
Paper : 111

Android Application Manager to Deactivate User Apps

Dr. Dhanakoti V [#1], Rahul Raj R[#2], Kasi Reddy Prakash[#3], Mohamad Hamdhan HM[#4]

[1]*Professor, Department of Computer Science and Engineering,SRM Valliammai Engineering College, Chennai, Tamil Nadu, India*
[2,3,4] *Department of Computer Science and Engineering, SRM Valliammai Engineering College, Chennai, Tamil Nadu, India*

Abstract— **Safety of the users information and management of the user security is the most important thing needed in today's android devices. We have developed an application using android studio with kotlin which enables the users to deactivate the apps after the usage of the application to ensure security measures and device performance. This project focus on providing a complete safety to the device by restricting the permissions granted for the application by deactivating the app.**

Keywords— **Android, Kotlin, Deactivate, Permissions, Security.**

I. INTRODUCTION

We may install more application for our day today's need. The most important point is when we download such application it requires certain permissions from the system settings which may include our personal data. After using the application we leave it alone when there is no need of it at that time the permission given to the app regarding the phone internal system may behave as threat to our private information. So by using our app there will be a complete protection over the application after when it is used by the user.

II. RELATED WORKS

Enck, et al. [1] have proposed a framework known as Kirin – install-time certification mechanism – that allows the mobile device to enforce a list of predefined security requirements prior to installation process of an application. During installation of an application the Android framework informs the user regarding the resources that can be accessed by the application but it cannot reflect the possibility of using different combinations of permission in a malicious manner.

Fuchs, et al. [2] proposed SCanDroid framework for Android to perform information flow analysis on applications in order to understand the flow of information from one component to another component. Consider a case where an application request permission to access multiple data stores i.e., public data store and private data store. The application requires permission for reading the data from the private store and writing data to the public store.

Ongtang, et al. [3] proposed Saint – a framework that provides security policy constraints in a more expressive manner by defining install-time permission granting policies and runtime policies for inter-component communication.

Conti, et al. [4] have proposed CRePE – a framework that enforces context-related fine-grained access control policies. It allows users to define policies that enable/ disable certain functionalities such as GPS, read SMS or Bluetooth discovery, based on the context of the device (e.g., location, noise, temperature, time, and nearby devices etc.).

Neha Verma, et al. [5] This paper describe about the Cab application developed using Android Studio new version. It also include about the Cross Platforms on which development of android and IOS both platforms application can be done. Cross platforms Paper also includes an example of Cab application which will show its working and its uses. Cab Application is an Android Application which is build in Android Studio 8.0.1

III. MATERIALS AND METHODS

A. Methods which are used to create this app which includes app class, packages and some other service has been used in this app to run as a whole. Each classes and service have a different functionalities and roles to satisfy

1) ActivityManager Class: The ActivityManager class provides methods to manage the activity lifecycle and other app related operations, including terminating background process. "killBackgroundProcess()" method is used to terminate the background process of the other apps

Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN : 2581 - 611X)
Volume 7- Issue 1, January 2024
Paper : 111

2) Android.content: This package contains classes and interfaces for accessing and publishing data on android. This package is used to retrieve information about installed packages and their processes

3) PackageManager Class: The PackageManager class provides methods to retrieve information about installed packages, including the package name and process name. We can use the "getInstalledPackages()" method to get a list of all installed packages, and then use the "killBackgroundProcess()" method to terminate the background processes of specific packages. This command helps to clear the background usage of the application which affects the performance of the mobile.

4) Android.os: This package contains classes for working with the operating system, including processes and threads.

5) Android.provider: This package provides access to the data stored in android's built in content providers, including information about installed apps.

6) Use Accessibility Services: Accessibility Services allow you to automate task on the device. You can use them to stimulate user input to navigate to the "force stop" button in the settings for the targeted app, which will terminate its process. However, this method requires the user to enable your app as an accessibility Service and also requires additional permissions.

B. Softwares and tools

1) Android Studio: Android Studio is the official integrated development environment (IDE) for Google's Android operating system, based on IntelliJ IDEA software from JetBrains and designed for Android development. It is available for download on Windows, macOS and Linux operating systems. It replaces the Eclipse Android Development Tools (E-ADT), the main IDE for native Android application development. Android Studio was announced at Google I/O on May 16, 2013. The first stable version was released in December 2014, starting with version 1.0. At the end of 2015, Google dropped support for Eclipse ADT, making Android Studio the only officially supported IDE for Android development.

*2) JavaDevelopmentKit(JDK):*The Java Development Kit (JDK) is Oracle Corporation's distribution of Java technology. It implements the Java Language Specification (JLS) and the Java Virtual Machine Specification (JVMS), and provides the Standard Edition (SE) of the Java Application Programming Interface (API). It is derived from the OpenJDK community managed by Oracle. [5] It provides software for working with Java applications. Examples of included software include virtual machines, compilers, performance monitoring tools, debuggers, and other utilities that Oracle deems useful to Java programmers.

*3) Android SDK (Software Development Kit):*It is a set of development tools for developing applications for theAndroid platform.The SDK provides all the tools needed to build Android apps and make sure the process runs as smoothly as possible. Whether you're building your app in Java, Kotlin, or C#, you need the SDK to run it on any Android device. You can also use the emulator to test the apps you build.Today the Android SDK also comes with Android Studio, the integrated development environment that does the work, and now has the best access or management of many tools.

C. Programming Language

Kotlin: Kotlin is a cross platform high level programming language with type inference. Kotlin is designed to work fully with java. The JVM version of the kotlin standard library depends on the java class library, but type inference makes its syntax more concise. Kotlin primarily targets the JVM, but is also available via LLVM(eg. For native ios apps that share business logic with android apps). Language development costs are broneby JetBeans, while Kotlin Foundation protects the kotlin brand.

IV. IMPLEMENTATION

The Android Studio is the platform we have used to develop our application in which we have used a programming language called kotlin which is efficient and better than java android coding. Firstly we have to design the user interface and all the other UI design related modules. Then install all the packages we need to use in the particular app development and initiate each package one by one according to the process which is going to occur. Our first function which is needed to be done is to list the number of applications which has been presented in the android mobile phone. Provide a background running app termination process using a background kill command and then turn off the network connectivity for that particular application. Now move the app package to the recovery folder and use a visibility mode command to hide the application from the mobile list menu. By following these implementations our android application can be built without any complications and each command can be modified

Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN : 2581 - 611X)

Volume 7- Issue 1, January 2024
Paper : 111

whenever we needed to change the functionality of the application.

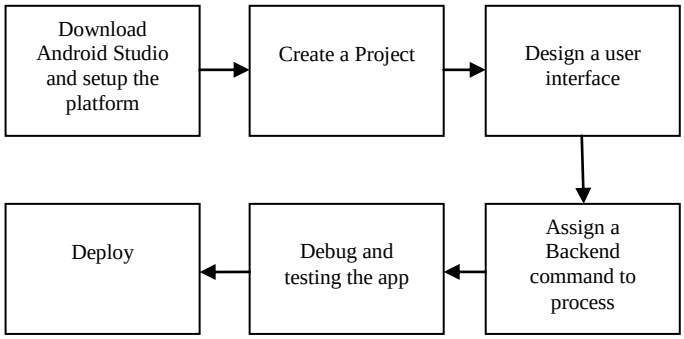
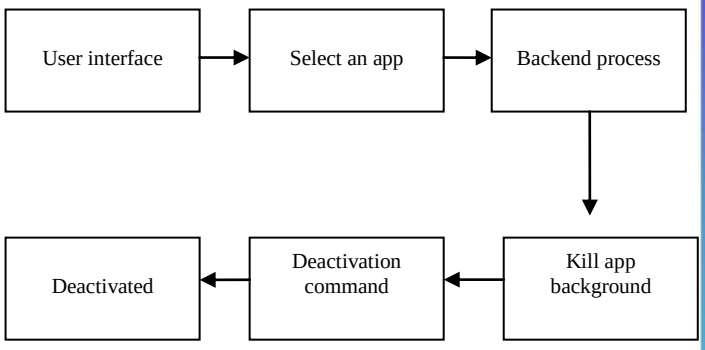
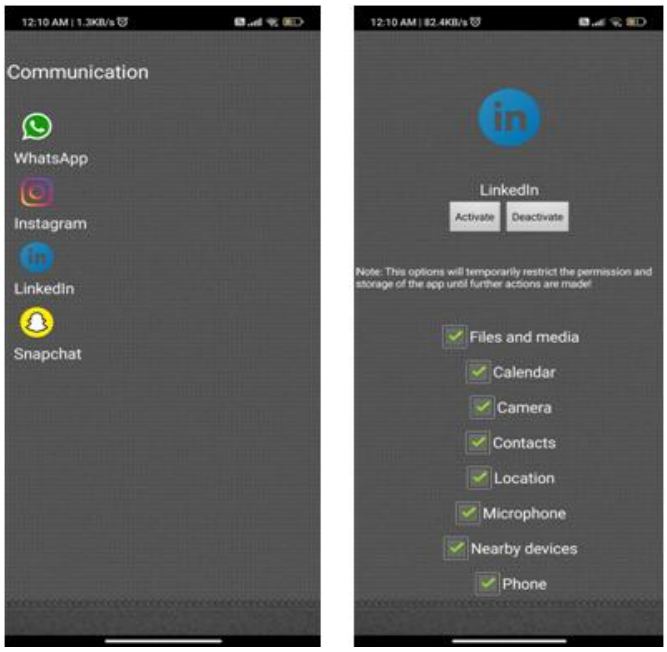


Fig. 1

Fig.1. Process to build an android app





Android Application Manager to Deactivate User Apps

Fig. 2

Fig.2. Process to deactivate application

A. User Interface Design

The user interface of the application has been designed using the **android studio** and **adobe xd** software where user interface is designed in a way where a user can easily classify their mobile application into separate category like games, entertainment, news, social media etc. The UI design also enables the user to have a access to see the all permissions granted by the user for each application.

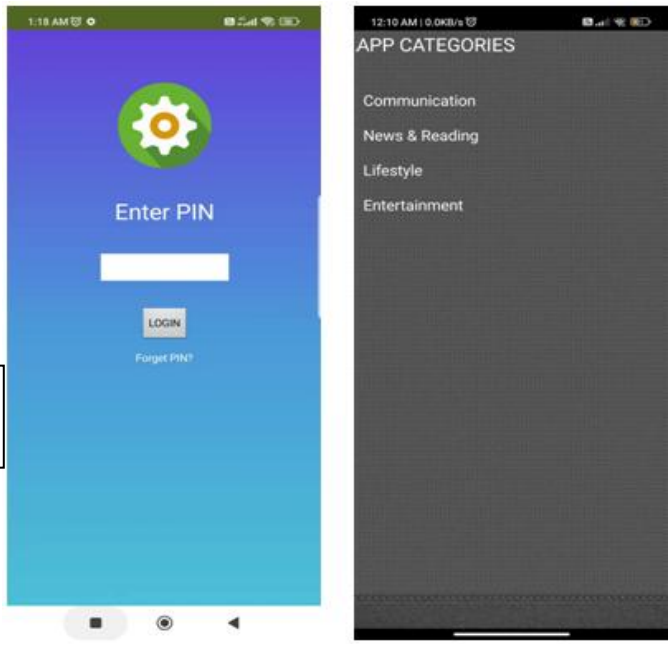


Fig. 3

Fig.3 User interface of the application

B. Backend command

The backend is the backbone of the application. After completion of the user interface design start to assign a backend command for the app. The backend has a several commands like retrieving the list of apps, background kill, disabling the app package etc. The following commands can

Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN : 2581 - 611X)

Volume 7- Issue 1, January 2024
Paper : 111

be executed to make the app fully deactivate from the mobile phone.

Add the necessary permissions: In app's manifest file, add the necessary permissions to disable other apps, such as the "android.permissions.PACKAGE_USAGE_STATS".

Disable apps: Use ActivityManager class to disable other apps. This can be done by calling the "setApplicationEnabledSetting" method with the "COMPONENT_ENABLE_STATE_DISABLE flag.

Fig. 4

Fig.4 Command to view list menu

Fig. 5

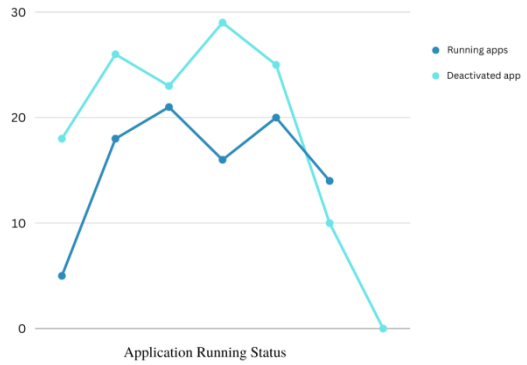
Fig.5 Command to disable the app package

Fig. 6

Fig.6 Command to kill the app background

V. Discussion

In this project starting from front end user interface to the backend processing has been done using android studio which is user friendly and genuine platform to develop an android apps which has been originally published by android. We have used the emerging android programming language called kotlin which is more efficient than the java android programming. To make it even more secure we have secured the app with user authentication security to use the app by the real user.



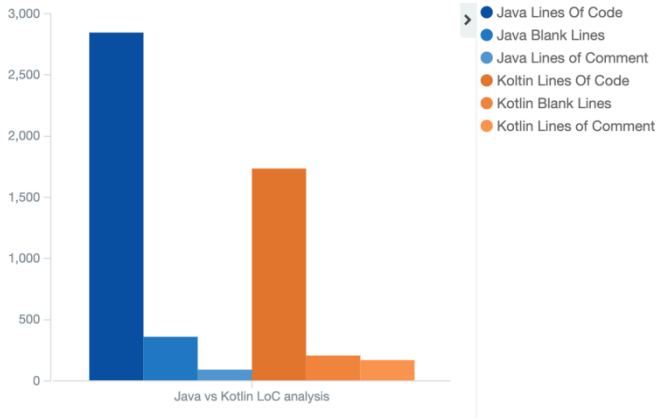


Fig. 7

Fig.7 Java vs Kotlin LoC analysis

```
PackageManager packageManager=getPackageManager();
List<ApplicationInfo>
apps=packageManager.getInstalledApplications(PackageManager.GET_META_DATA);
```

VI. Result

The final result of this project is that the selected user application has been successfully deactivated and it made to

```
ComponentName componentName =  new ComponentName("com.example.appname",
"com.example.appname.MainActivity");
PackageManager packageManager = getPackageManager();
packageManager.setComponentENabledSetting(componentName,
PackageManager.COMPONENT_ENABLED_STATE_DISABLED,
PackageManager.DONT_KILL_APP);
```

restrict all the permissions which has been granted during the app installation which also prevents from third party application hacking. The deactivation has done with uninstalling the application or clearing the user data.

```
ActivityManager activityManager = (ActivityManager)
getSystemService(Context.ACTIVITY_SERVICE);
List<ActivityManager.RunningAppProcessInfo processInfo : runningProcess)
{
 if(processInfo.importance ==
ActivityManager.RunningAppProcessInfo.IMPORTANCE_BACKGROUND)
{
  String packageName = processInfo.processName;
  activityManager.killBackgroundProcess(packageName);
}
}
```

Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN : 2581 - 611X)
Volume 7- Issue 1, January 2024
Paper : 111

Fig. 8

Fig.8 Status of the application in the mobile

The above graph shows the frequency graph of the running application. The frequency of the application has suddenly drop down after the application has been deactivated.

VII. Conclusion

This android application manager would bring the drastic change in the android platform system. The created model helps in improving security features of the mobile which includes all the private data and by deactivating the apps will result in better performance and restrict the permission and user don't need to worry about security attacks happen due to third party application. This can be done without loosing the application data and uninstalling the app. This app will surely be a user friendly and easy to use by all the android users.

References

[1] W. Enck, D. Octeau, P. Mcdaniel, and S. Chaudhuri. A study of android application security. USENIX Security, (August):935– 936, 2011. http://www.usenix.org/event/sec11/tech/slides/enck.pdf.

[2] A. Fuchs, A. Chaudhuri, and J. Foster, "Scandroid: Automated security certification of android applications," Manuscript, Univ. of Maryland, http://www.cs.umd.edu/˜avik/projects/scandroidasc aa.

[3] M. Ongtang, K. Butler, and P. McDaniel, "Porscha: Policy oriented secure content handling in android," in Proceedings of the 26th Annual Computer Security Applications Conference. ACM, 2010, pp. 221–230.

[4] M. Conti, V. Nguyen, and B. Crispo, "Crepe: Context-related policy enforcement for android," Information Security, pp. 331–345, 2011.

[5] Neha Verma, Sarita Kansal and Huned Malvi,"Development of Native Mobile Application Using Android Studio for Cabs and Some Glimpse of Cross Platform," International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 16 (2018) pp. 12527-12530.