Al Imam Mohammad Ibn Saud Islamic University
College of Computer and Information Sciences
Computer Science Department

CS292 Software Engineering2

Quality Assessment report

(Riyadh Bank App)

Section:373

Names	IDs	emails
Ghadeer Muiedh	4400204093	gmhalshalawi@sm.imamu.edu.sa
Razan Bajaaman	440022262	raabajaaman@sm.imamu.edu.sa
Hanan Almutairi	440022269	howalmutairi@sm.imamu.edu.sa

Supervisor Name: Dr. Lamees Alhazzaa

Contents:

1. Introduction

1.1 Purpose:

The first workshop purpose is to introduce the empirical studies on the quality assessments of local banking apps to provide useful insights and improve the quality of these apps

1.2 Goal:

The goal of the workshop is to find the applications issues in terms of code quality and security, application design. In addition, provide useful solutions and insights and improve the quality of this apps.

1.3 Lessons learned:

At first sight we thought that bank applications may be free from problems and defects. However, the reality was that there were security problems and problems that might confuse developers due to the lack of documentation of the code and other problems.

We also discovered the existence of tools that help in the process of quality assessment, such as **Mobsf** and **GitHub codacy**[1] which facilitated our work a lot.

1.4 References

- [1] "Codacy", *GitHub*, 2021. [Online]. Available: https://github.com/marketplace/codacy.
- [2] Owasp.org. 2021. *OWASP Foundation | Open Source Foundation for Application Security*. [online] Available at: <https://owasp.org/>
- [3] Cwe.mitre.org. 2021. *CWE - Common Weakness Enumeration*. [online] Available at: <https://cwe.mitre.org>

2. Mobile App analysis results:

2.1 The steps to extract the mobile app data:

1- The first step is to download the APK of the application from APK downloader site.

2- we uploaded the APK to **MobSF** tool to start the static analysis.

3- Through the tool, we were able to download the source code.

4- After checking the code folders, we found the Design Pattern folder used in the application, the folder under “lifecycle" name.

5- Also, through the tool, we obtained an analysis of some system problems, such as security problems etc.

6- and We created a GitHub account to take advantage of some of the site's extensions to check the Quality code.

7- We downloaded the Riyad Bank application to check the design of the interfaces and the usability of the application from the user's perspective.

2.2 The issues we found:

1- We had a hard time downloading the APK for the app because of the copyrights of the bank even though it is supposed to be open source ,But after trying 7 download sites, we were able to download it successfully

2- It was also difficult to download the tool from the first time, as we faced almost the same problem that the download takes time and then stops in the middle, so we must delete it and download it again, but in the end, we were able to download it successfully

3- At first, we had a problem understanding the results of the tool’s analysis, but after watching some videos and researching, we were able to understand and use them in writing the report.

3. Analysis Result Discussion:

3.1 Security:

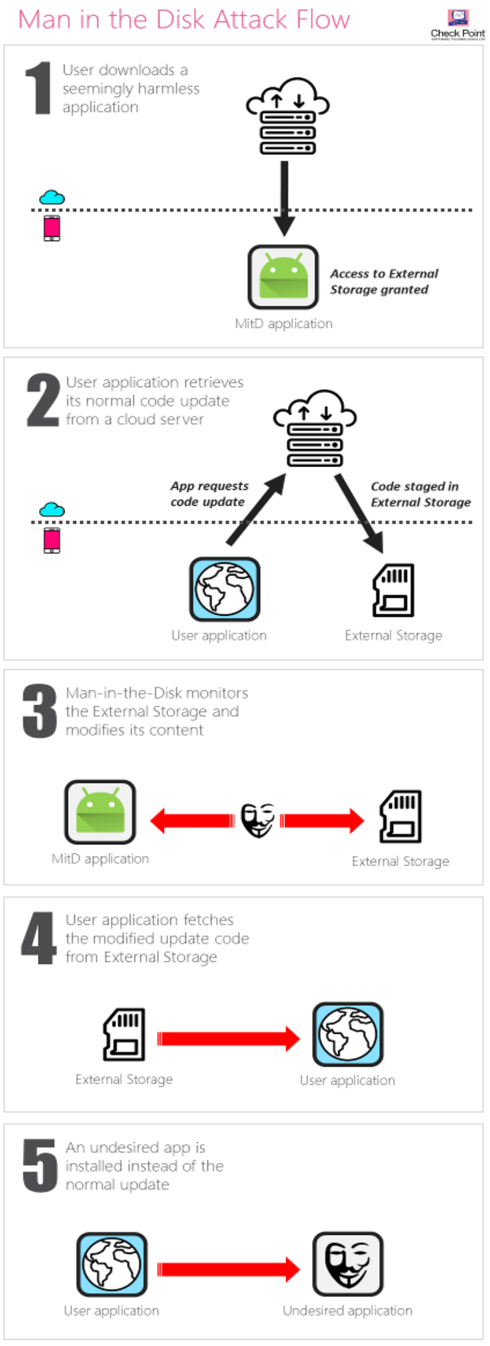
3.1.1 Code analysis:

After examining the application code, we found some security problems that may threaten the safety of users and their data, and we have provided some solutions that may contribute to raising the level of security for the application and solving these problems described below:

	issue	**priority**	**explain**	**Recommendation**
1	App can read and write on external storge	high	It's may lead to ***man-in-disk*** *attack (illustrated in figure 1,2)* which can cause manipulation and \or abuse of unprotected sensitive data.	Try to use internal storage only, or follow Android's guidelines for external storage, which contain some steps that help reduce the risks of using external storage.
2	Using SHA-1 as cryptographic algorithm	high	It's weak algorithm for hashing and insufficient Which makes the application less secure.	Verify that cryptographic algorithms are up to date and in-line with industry standards, Note that even algorithms that are certified (for example, by NIST) can become insecure over time so you should do periodic verification of an algorithm's soundness. also make sure that Algorithms with known weaknesses should be replaced with more secure alternatives and you can check owasp.com for more information.
3	The App uses an insecure Random Number Generator	high	The use of weak random number generators can lead to prediction attacks, which in turn threaten the security of the application.	use Cryptographically secure RNGs generate random numbers that pass statistical randomness tests, and are resilient against prediction attacks (e.g., it is statistically infeasible to predict the next number produced).

4	App creates temp file. Sensitive information should never be written into a temp file	Medium	Insecure Data Storage because creating a temporary file allows all user access to it, which may be easily accessible to attackers, possibly enabling them to read and modify the contents of the temp file.	you can reduce the risks by doing the following: 1- Ensure that you use proper file permissions. This can be achieved by using a safe temp file function. Temporary files should be writable and readable only by the process that owns the file. 2- Randomize temporary file names. This can also be achieved by using a safe temp-file function. This will ensure that temporary files will not be created in predictable places.
5	The App logs information. Sensitive information should never be logged.	high	logging sensitive data may expose the data to attackers or malicious applications, and it might also violate user confidentiality	Consider seriously the sensitivity of the information written into log files. Do not write sensitive data into the log files and for more information you can check https://cwe.mitre.org/data/definitions/532.html
6	Calling Cipher.getInstance("AES") will return AES ECB mode by default. ECB mode is known to be weak as it results in the same ciphertext for identical blocks of plaintext.	Medium	we don't have the experience about this subject, but it could be a problem based on The Open Web Application Security Project (OWASP) and you can learn more about the problem visiting: https://github.com/MobSF/owasp-mstg/blob/master/Document/0x04g-Testing-Cryptography.md#weak-block-cipher-mode	we don't have the experience about this subject, but The Open Web Application Security Project (OWASP) recommended some solution that help you avoid this problem you can check it through the link below: https://github.com/MobSF/owasp-mstg/blob/master/Document/0x04g-Testing-Cryptography.md#weak-block-cipher-mode

7	The App uses the encryption mode CBC with PKCS5/PKCS7 padding. This configuration is vulnerable to padding oracle attacks.	Medium	we don't have the experience about this subject, but it could be a problem based on The Open Web Application Security Project (OWASP) and you can learn more about the problem visiting: https://github.com/MobSF/owasp-mstg/blob/master/Document/0x04g-Testing-Cryptography.md#identifying-insecure-andor-deprecated-cryptographic-algorithms-mstg-crypto-4	we don't have the experience about this subject, but The Open Web Application Security Project (OWASP) recommended some solution that help you avoid this problem you can check it through the link below: https://github.com/MobSF/owasp-mstg/blob/master/Document/0x04g-Testing-Cryptography.md#identifying-insecure-andor-deprecated-cryptographic-algorithms-mstg-crypto-4
8	This App copies data to clipboard. Sensitive data should not be copied to clipboard as other applications can access it.	Low	The clipboard is accessible system-wide and is therefore shared by apps. This sharing can be misused by malicious apps to get sensitive data that has been stored in the clipboard.	Where appropriate, disable copy/paste for areas handling sensitive data. In addition, it can be interesting to clear the clipboard after taking the contents, to avoid other apps read them and leak what the user is doing.
9	Files may contain hardcoded sensitive information like usernames, passwords, keys etc.	high	we don't have the experience about this subject, but it could be a problem based on The Open Web Application Security Project (OWASP) and you can learn more about the problem visiting: https://github.com/MobSF/owasp-mstg/blob/master/Document/0x05d-Testing-Data-Storage.md#checking-memory-for-sensitive-data-mstg-storage-10	We don't have the experience about this subject, but The Common Weakness Enumeration (CWE) recommended some solution that help you avoid this problem you can check it through the link below: https://cwe.mitre.org/data/definitions/798.html


Man in the Disk Attack Flow
Check Point
1 User downloads a seemingly harmless application
Access to External Storage granted
MitD application
2 User application retrieves its normal code update from a cloud server
App requests code update
Code staged in External Storage
User application
External Storage
3 Man-in-the-Disk monitors the External Storage and modifies its content
MitD application
External Storage
4 User application fetches the modified update code from External Storage
External Storage
User application
5 An undesired app is installed instead of the normal update
User application
Undesired application

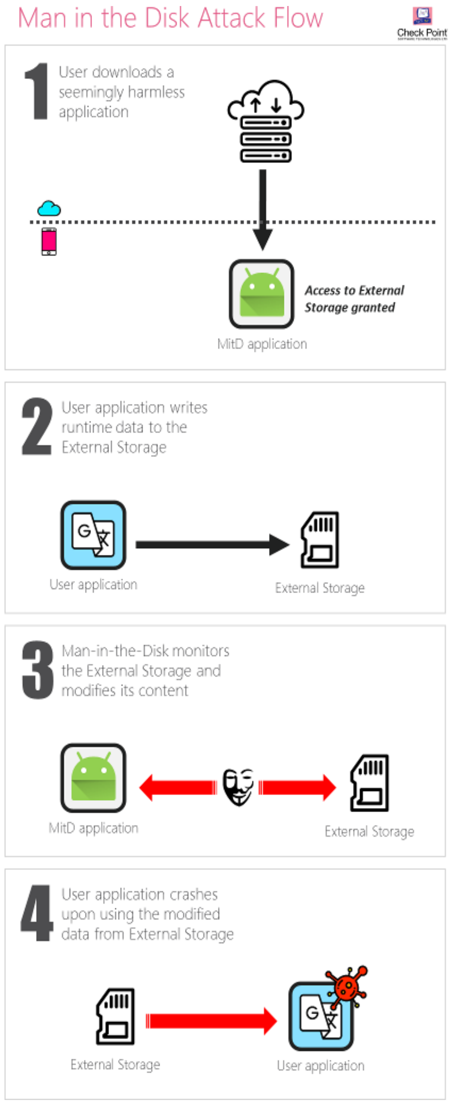

Man in the Disk Attack Flow
Check Point
1 User downloads a seemingly harmless application
Access to External Storage granted
MitD application
2 User application writes runtime data to the External Storage
User application
External Storage
3 Man-in-the-Disk monitors the External Storage and modifies its content
MitD application
External Storage
4 User application crashes upon using the modified data from External Storage
External Storage
User application

Figure 1,2: illustrative images for an attack "Man in The Disk"

Assess the severity of problems based on CVSS:

In the chart below, an analysis of the severity of the previous problems based on Common Vulnerability Scoring System (CVSS) which provides a way to capture the principal characteristics of a vulnerability and produce a numerical score reflecting its severity

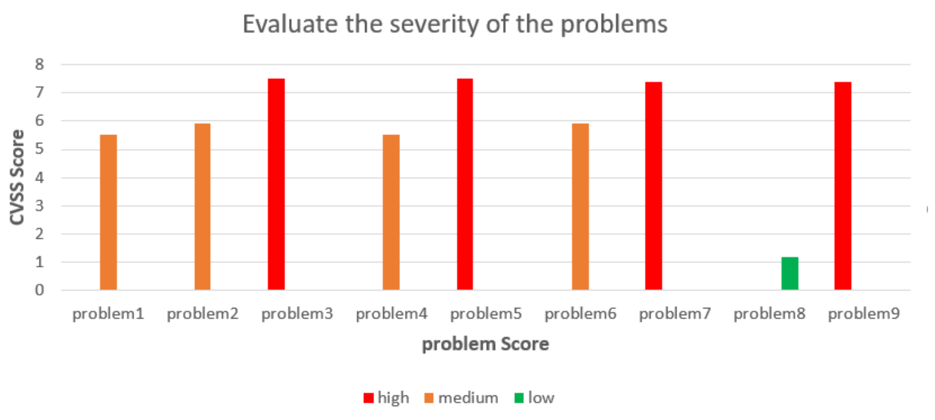


Figure2: Evaluate the severity of problems based on CVSS

3.1.2 Other security Problems:

We have noticed that there are a lot of Share services, activities and objects that allow the rest of the applications on the device to access it and thus be exposed to any possible attack.

Recommendation:
Other applications must be prevented from accessing application resources such as objects activities and services completely.

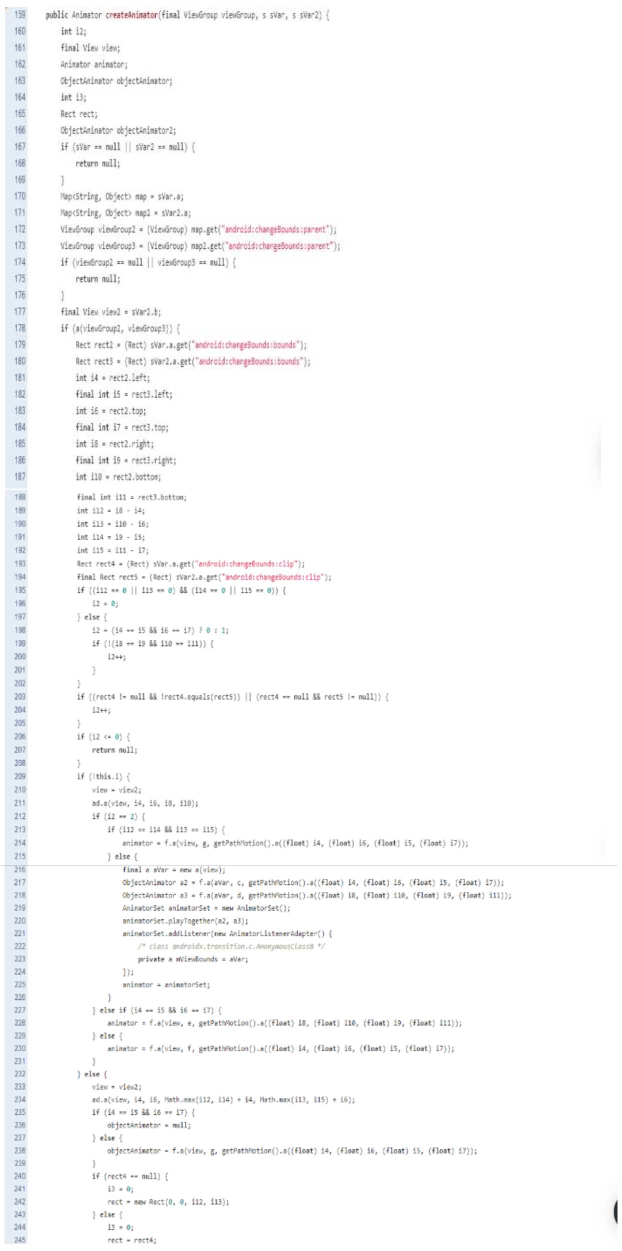
but in case that the application needs to share these resources, permissions must be set and approved by the user and explain in it about the risks of sharing these resources and why the application needs it.

3.2 Application Code Quality:

1- Too long method

Explain:
When methods are excessively long this usually indicates that the method is doing more than its name/signature might suggest Therefore, it violates the *Single responsibility principle*. They also become challenging for others to digest since excessive scrolling causes readers to lose focus.

Recommendation:
try to reduce the method length by creating helper methods and removing any copy/pasted code.



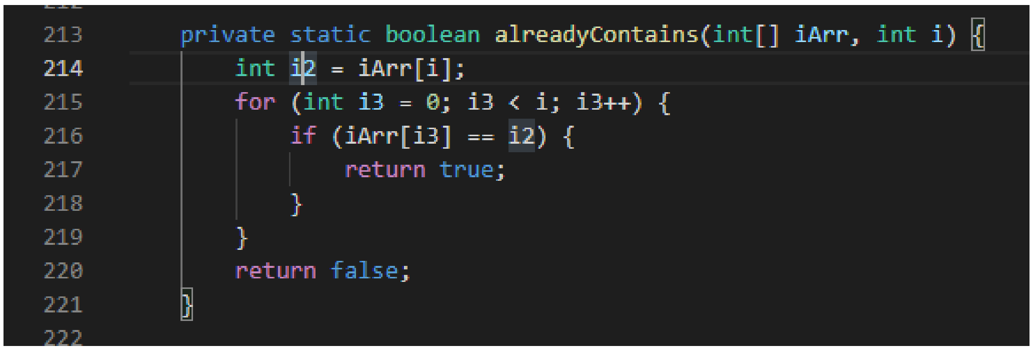
```java
public Animator createAnimator(final ViewGroup viewGroup, s sVar, s sVar2) {
    int i2;
    final View view;
    Animator animator;
    ObjectAnimator objectAnimator;
    int i3;
    Rect rect;
    ObjectAnimator objectAnimator2;
    if (sVar == null || sVar2 == null) {
        return null;
    }
    Map<String, Object> map = sVar.a;
    Map<String, Object> map2 = sVar2.a;
    ViewGroup viewGroup2 = (ViewGroup) map.get("android:changeBounds:parent");
    ViewGroup viewGroup3 = (ViewGroup) map2.get("android:changeBounds:parent");
    if (viewGroup2 == null || viewGroup3 == null) {
        return null;
    }
    final View view2 = sVar2.b;
    if (a(viewGroup2, viewGroup3)) {
        Rect rect2 = (Rect) sVar.a.get("android:changeBounds:bounds");
        Rect rect3 = (Rect) sVar2.a.get("android:changeBounds:bounds");
        int i4 = rect2.left;
        final int i5 = rect3.left;
        int i6 = rect2.top;
        final int i7 = rect3.top;
        int i8 = rect2.right;
        final int i9 = rect3.right;
        int i10 = rect2.bottom;
        final int i11 = rect3.bottom;
        int i12 = i8 - i4;
        int i13 = i10 - i6;
        int i14 = i9 - i5;
        int i15 = i11 - i7;
        Rect rect4 = (Rect) sVar.a.get("android:changeBounds:clip");
        final Rect rect5 = (Rect) sVar2.a.get("android:changeBounds:clip");
        if ((i12 == 0 || i13 == 0) && (i14 == 0 || i15 == 0)) {
            i2 = 0;
        } else {
            i2 = (i4 == i5 && i6 == i7) ? 0 : 1;
            if (!(i8 == i9 && i10 == i11)) {
                i2++;
            }
        }
        if ((rect4 != null && !rect4.equals(rect5)) || (rect4 == null && rect5 != null)) {
            i2++;
        }
        if (i2 <= 0) {
            return null;
        }
        if (!this.i) {
            view = view2;
            ad.a(view, i4, i6, i8, i10);
            if (i2 == 2) {
                if (i12 == i14 && i13 == i15) {
                    animator = f.a(view, g, getPathMotion().a((float) i4, (float) i6, (float) i5, (float) i7));
                } else {
                    final a aVar = new a(view);
                    ObjectAnimator a2 = f.a(aVar, c, getPathMotion().a((float) i4, (float) i6, (float) i5, (float) i7));
                    ObjectAnimator a3 = f.a(aVar, d, getPathMotion().a((float) i8, (float) i10, (float) i9, (float) i11));
                    AnimatorSet animatorSet = new AnimatorSet();
                    animatorSet.playTogether(a2, a3);
                    animatorSet.addListener(new AnimatorListenerAdapter() {
                        /* class androidx.transition.c.AnonymousClass8 */
                        private a mViewBounds = aVar;
                    });
                    animator = animatorSet;
                }
            } else if (i4 == i5 && i6 == i7) {
                animator = f.a(view, e, getPathMotion().a((float) i8, (float) i10, (float) i9, (float) i11));
            } else {
                animator = f.a(view, f, getPathMotion().a((float) i4, (float) i6, (float) i5, (float) i7));
            }
        } else {
            view = view2;
            ad.a(view, i4, i6, Math.max(i12, i14) + i4, Math.max(i13, i15) + i6);
            if (i4 == i5 && i6 == i7) {
                objectAnimator = null;
            } else {
                objectAnimator = f.a(view, g, getPathMotion().a((float) i4, (float) i6, (float) i5, (float) i7));
            }
            if (rect4 == null) {
                i3 = 0;
                rect = new Rect(0, 0, i12, i13);
            } else {
                i3 = 0;
                rect = rect4;
```

Figure3: createAnimator method

2- No validation

Explain:
This can cause a problem if the variables are not validated before they are used because they may be null, which leads to an error in the working of the system, or it may become an error affected by it and often this type of error cannot be detected easily.

Recommendation:
Therefore, it is recommended to put a conditional statement to ensure that these variables will not become a threat to the system.



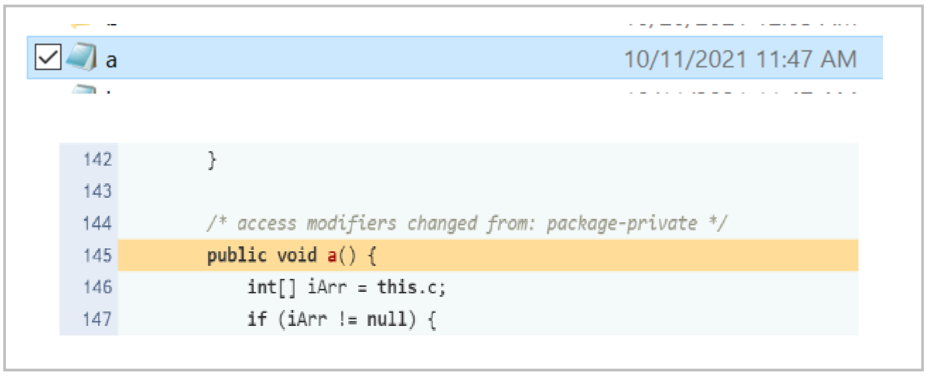
```java
    private static boolean alreadyContains(int[] iArr, int i) {
        int i2 = iArr[i];
        for (int i3 = 0; i3 < i; i3++) {
            if (iArr[i3] == i2) {
                return true;
            }
        }
        return false;
    }
```

Figure4: Method contains values that are not validated before use

3- Method with same name as enclosing class

Explain:
It is allowed to define a method with the same name as that of a class. There is no error will occur. But this is not recommended as per coding standards in Java. Normally the constructor's name and class name always the same in Java.

Recommendation:
Generally, a class's name should be a noun, while a method's name should be a verb, To be clear to the developer and can differentiate between them.



a 10/11/2021 11:47 AM

```java
    }

    /* access modifiers changed from: package-private */
    public void a() {
        int[] iArr = this.c;
        if (iArr != null) {
```

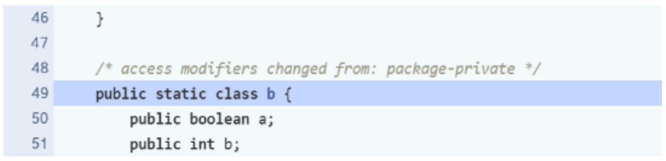
Figure5: Method with same name as enclosing class

4- class naming conventions:

Explain:
It's a coding convention, adopted by most Java programs. It makes reading code easier as you become use to a given standard.

Recommendation:
It's nicer when everyone writes code in a similar way. Makes it easier to understand other people's code, It's the same thing with natural languages, like English. yOU CaN bREak thE RuLeS, tHE mEsSaGe GETS tHrouGH anYwaY, but iT lOOk vErY sillY.

Try to keep your class names simple and descriptive, If you are naming any class then it should be a noun and so should be named as per the goal to be achieved in the program such as MaxNumbers, CountWord, and so on not likely A, A1, Programming, etc.



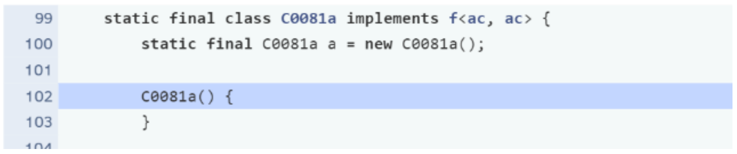
```
    }

    /* access modifiers changed from: package-private */
    public static class b {
        public boolean a;
        public int b;
```

Figure6: class naming conventions.

5- Unnecessary constructer:

Explain:
We can avoid unnecessary constructer if we need it the compiler will generate one, here we don’t now maybe we don’t need it so that will take some space without any reason.

Recommendation:
In some cases, it is unnecessary as demonstrated in this example and in some cases it is necessary. super() will be automatically inserted. You can just leave it to insert automatically to avoid unnecessary constructer.



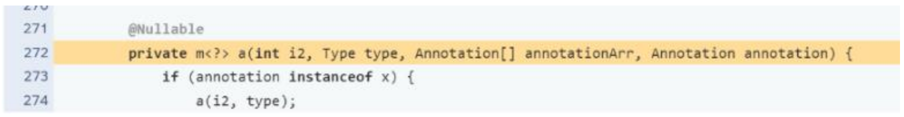
```
    static final class C0081a implements f<ac, ac> {
        static final C0081a a = new C0081a();

        C0081a() {
        }
```

Figure6: unnecessary constructer.

6- Unused private method:

Explain:
The private method is not accessible, which leads to not using it and taking up space without any benefit.

Recommendation:
private methods that are never executed are unnecessary that should be removed. Clean unused code making it easier to understand the program and preventing bugs from being introduced.



```
        @Nullable
        private m<?> a(int i2, Type type, Annotation[] annotationArr, Annotation annotation) {
            if (annotation instanceof x) {
                a(i2, type);
```

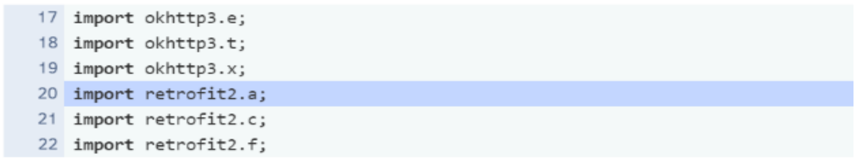
Figure7: unused private method

7- Unused Import:

Explain:
If another developer is searching through code for something, it's a pain to look through your code and discover what they were looking for was a false flag.

Your unused import could have naming conflicts with another import in this case that is actually being used. The annoying warning sign is one thing, however, forcing yourself to write a different name simply to fix this import error.

Recommendation:
Try to write the imports when you need it don’t write it all in the first you may fall on the same mistake.



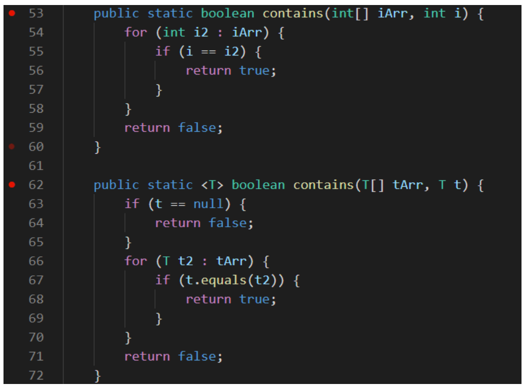
```java
import okhttp3.e;
import okhttp3.t;
import okhttp3.x;
import retrofit2.a;
import retrofit2.c;
import retrofit2.f;
```

Figure8: unused Import

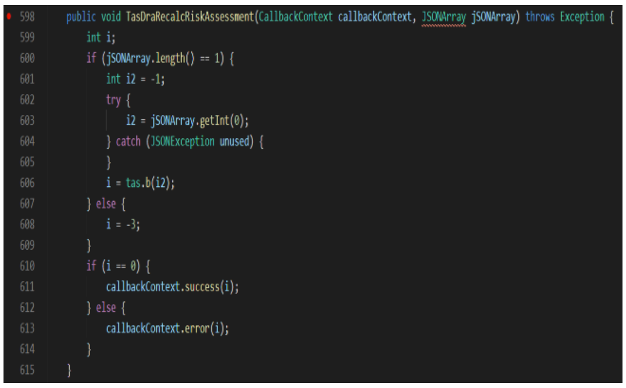
8- Meaningful identifiers:

Explain:
non-meaningful identifiers is ineffective because there is not any programmer can remember what they have Code 7-8 Months Ago for example and when they will see that source code again, it will be difficult to Understand.

Recommendation:
Meaningful identifiers help the programmers to understand the Code Written Long Time Ago and it is self-documentation, ex:NumberSum().



```java
public static boolean contains(int[] iArr, int i) {
    for (int i2 : iArr) {
        if (i == i2) {
            return true;
        }
    }
    return false;
}

public static <T> boolean contains(T[] tArr, T t) {
    if (t == null) {
        return false;
    }
    for (T t2 : tArr) {
        if (t.equals(t2)) {
            return true;
        }
    }
    return false;
}
```



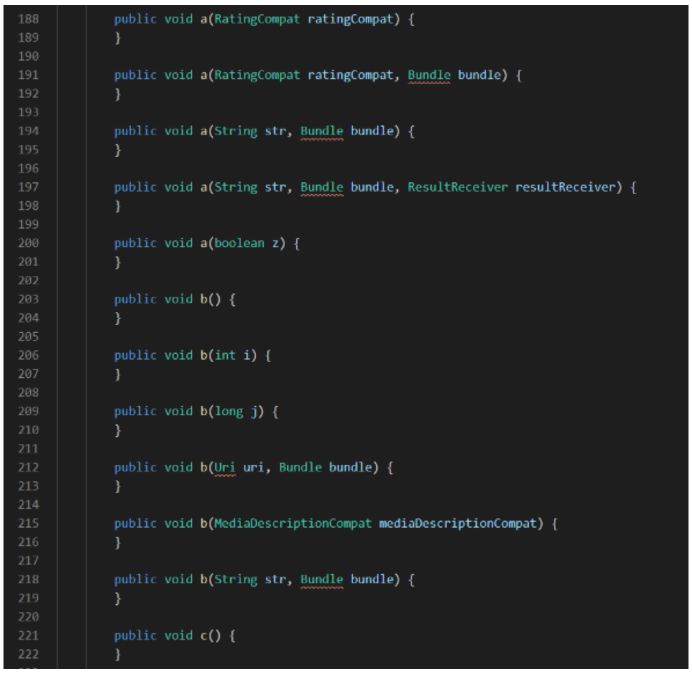
```java
public void TasDraRecalcRiskAssessment(CallbackContext callbackContext, JSONArray jSONArray) throws Exception {
    int i;
    if (jSONArray.length() == 1) {
        int i2 = -1;
        try {
            i2 = jSONArray.getInt(0);
        } catch (JSONException unused) {
        }
        i = tas.b(i2);
    } else {
        i = -3;
    }
    if (i == 0) {
        callbackContext.success(i);
    } else {
        callbackContext.error(i);
    }
}
```

Figure9,10: images of some methods with meaningless or unclear names

10- Empty method body

Explain:
Not to write a comment in the empty method. maybe he forgot to insert the body, or it is intentional, and we can't know if it was intentional or not.

Recommendation:
Write a comment explaining why it is empty or why it is used and whether it works or not.



```java
    public void a(RatingCompat ratingCompat) {
    }

    public void a(RatingCompat ratingCompat, Bundle bundle) {
    }

    public void a(String str, Bundle bundle) {
    }

    public void a(String str, Bundle bundle, ResultReceiver resultReceiver) {
    }

    public void a(boolean z) {
    }

    public void b() {
    }

    public void b(int i) {
    }

    public void b(long j) {
    }

    public void b(Uri uri, Bundle bundle) {
    }

    public void b(MediaDescriptionCompat mediaDescriptionCompat) {
    }

    public void b(String str, Bundle bundle) {
    }

    public void c() {
    }
```

Figure12: An empty methods and without comments explaining its purpose

11- Documentation code:

Explain:
Not writing documentation and explaining what the code does and what is the algorithm etc., which makes the code difficult to understand and a waste of time, also, the programmer may forget what methods do/what is the algorithm used etc., if a long-time passed.

Recommendation:
writing documentation and explain what the method/function broadly does and what it expects from the inputs and from the outputs and code not clear when reading it, Documentation provides more understanding and facilitates the maintenance process and improves the quality of the software product.

3.3 Application Design:

3.3.1 Application Structural design:

In terms of the Bank's Design Patterns, they used the composite pattern, the Observer Pattern and the Adapter Pattern and from our perspective:

The observer pattern good since it’s reducing the coupling but we do not think that the bank app needs The Observer pattern because this pattern supports the server and client system, so that the client asks the server for notifications about anything it is interested in, but the bank does not need to send notifications such as discounts, advertisements, etc., when the bank needs something from the client It sends the request in particular to the required client, such as updating the national identity. Not everyone needs to be updated and they do not need a subscription to get these updates. The bank is responsible for who receives the message, not the client.

And We noticed that they used the strategy pattern, and it was not present in the source code, The app depends on the strategic pattern as shown in the Figure13, so that the customer can choose the appropriate method for him to complete his work.

Recommendation:

We suggest using the Decorator Pattern because the bank system needs a pattern that supports the updates and the changes in the code, It helps in applying updates or changes to a part of the system without changing or affecting the entire system

Also, we suggest using the facade pattern, because the client needs an easy interface that hides the complex details of the system.

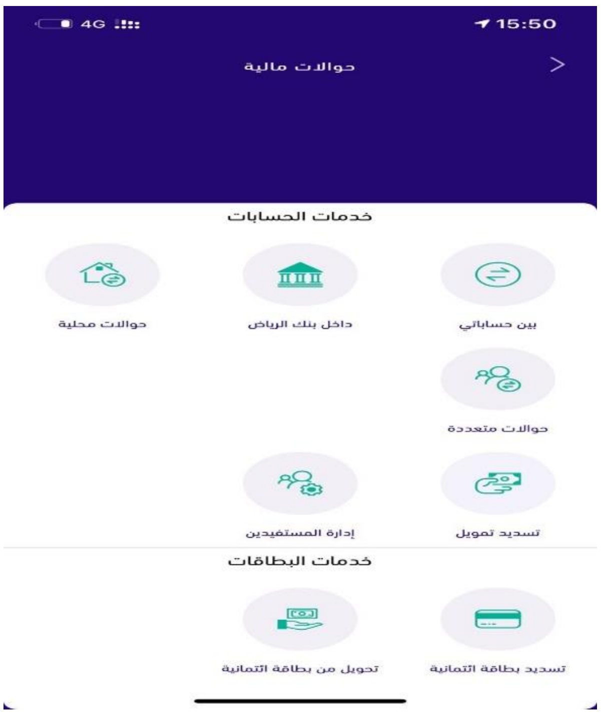
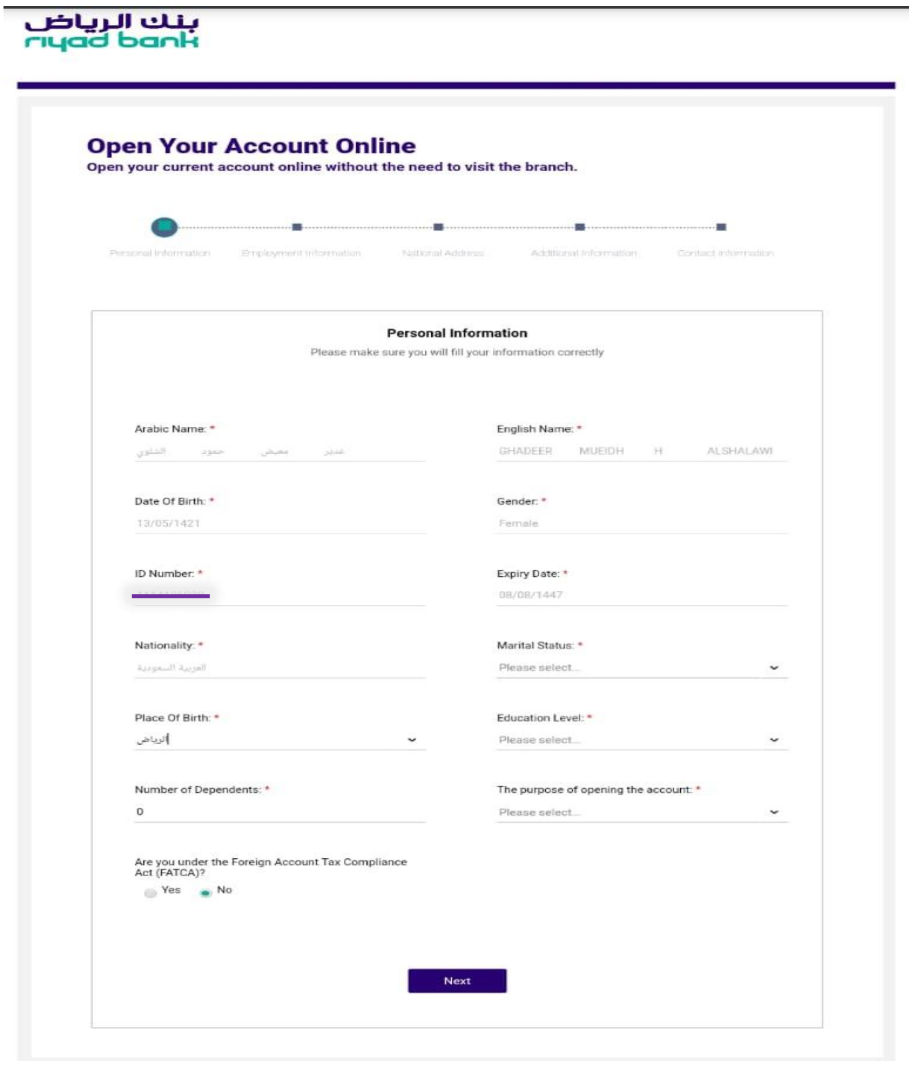


Figure 13: Application home page

3.3.2 Application interface design:

1- **There is no Arabic interface:**
We tried to register a new account using the app and were redirected to the webpage where there is an interface that supports English only as shown in the picture and this may affect the usability of the app

Recommendation:
As a Saudi bank application, it is very important that all interfaces support the Arabic language



Open Your Account Online
Open your current account online without the need to visit the branch.

Personal Information
Please make sure you will fill your information correctly

Arabic Name: *	English Name: *
الشلوي حمود معيض غدير	GHADEER MUEIDH H ALSHALAWI
Date Of Birth: *	Gender: *
13/05/1421	Female
ID Number: *	Expiry Date: *
[redacted]	08/08/1447
Nationality: *	Marital Status: *
العربية السعودية	Please select...
Place Of Birth: *	Education Level: *
الرياض	Please select...
Number of Dependents: *	The purpose of opening the account: *
0	Please select...

Are you under the Foreign Account Tax Compliance Act (FATCA)?
Yes [✓ No]

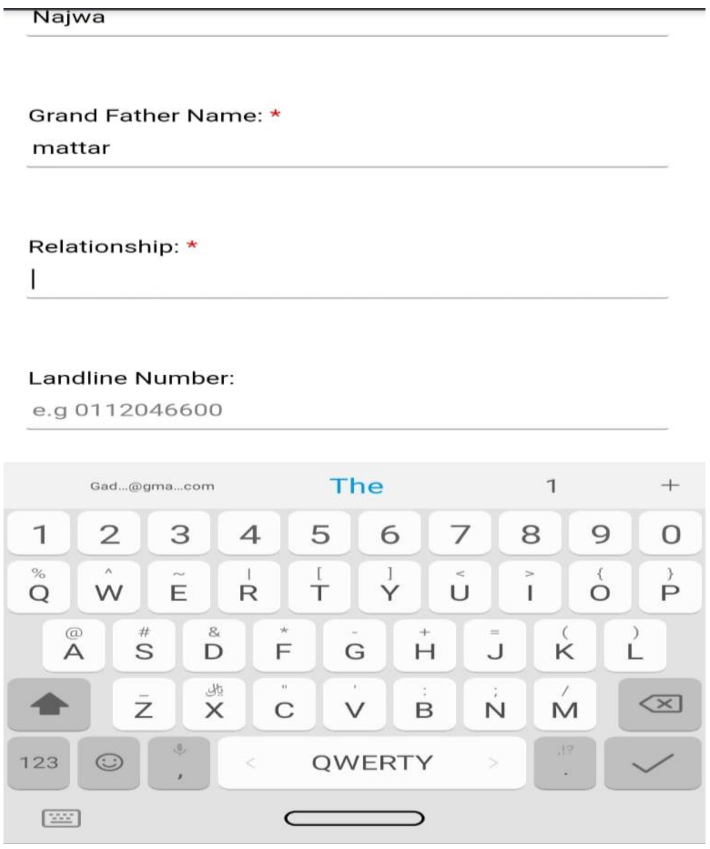
Figure 14: Account registration page

2- better to use selected list :

As shown in the picture, Field was used to write the type of kinship, which may allow the user to write his own As shown in the picture, Field was used to write the type of kinship, which may allow the user to write his own vocabulary as an answer to this question instead of the usual formal vocabulary for this type of question and also may affect the usability of the application as an answer to this question instead of the usual formal vocabulary for this type of question.

Recommendation:

It is preferable to use the *selected list* for this type of questions, which helps to restrict the user's choices and thus increase the formality in filling out his data and increase the usability of the application.



Najwa

Grand Father Name: *
mattar

Relationship: *

Landline Number:
e.g 0112046600

Figure 15: Account registration page

3- Unimportant or redundant questions that may annoy the user:

When we try registered as new users, one of the requests to be filled in was a friend's data.

it can be a annoying for the users for some reasons below:

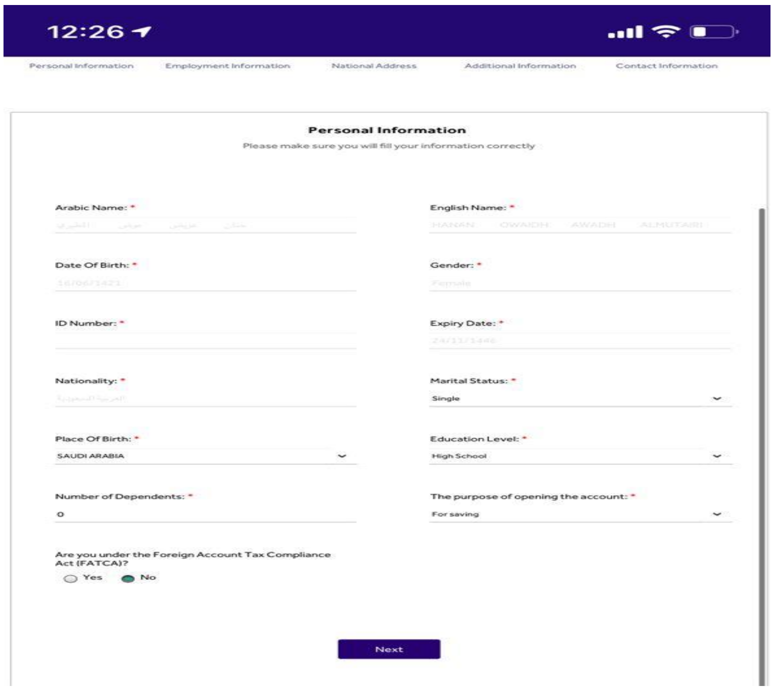
1) The user may not have all the personal information of his friend.

2) Asking the user to his friend about his personal information may be embarrassing to him.

Also, when registering, we entered an ABSHER account, and here it was supposed to be able to obtain personal information from ABSHER system, but it asked me to enter some non-important information that it could obtain it from ABSHER system, such as Marital status and the numbers of dependents Although it could obtain my full name, national identity, and national address.

Recommendation:

Reducing questions that may bother users and alienate them from using the application, such as asking them about the personal information of their friends. When using the Absher system, most of the information can be taken by the bank from this system, and this helps in facilitating the registration procedures by reducing questions.


12:26

Personal Information | Employment Information | National Address | Additional Information | Contact Information

Personal Information

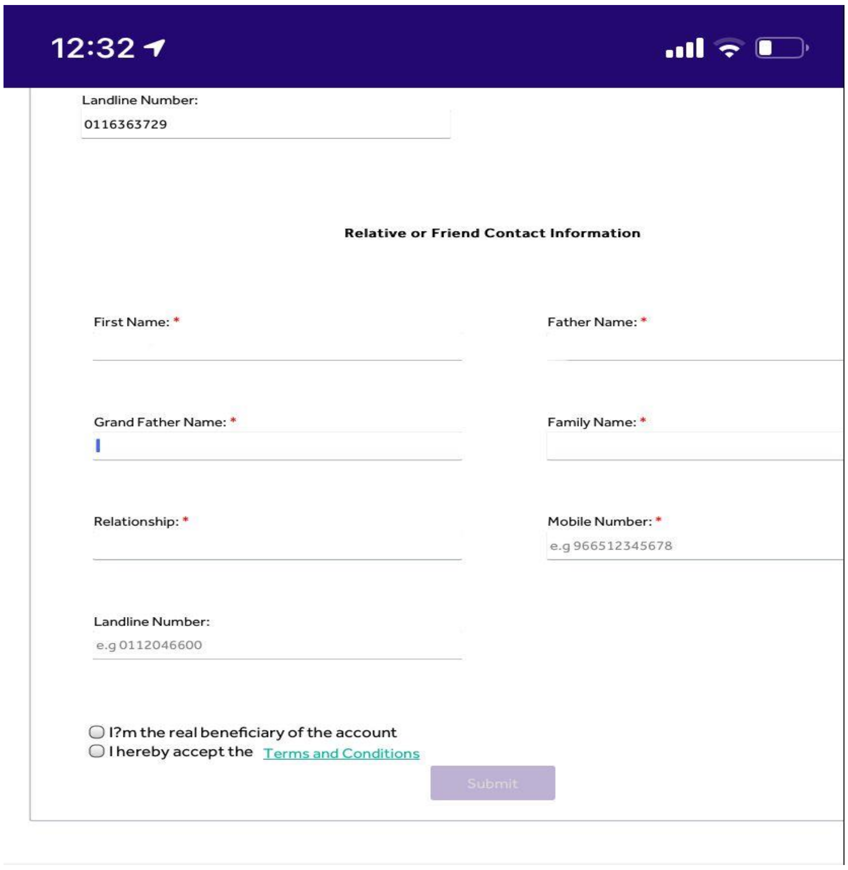
Please make sure you will fill your information correctly

Arabic Name: * [illegible]	English Name: * HANAN OWAIDH AWADH ALMUTAIRI
Date Of Birth: * 16/06/1421	Gender: * Female
ID Number: *	Expiry Date: * 24/11/1446
Nationality: * العربية السعودية	Marital Status: * Single
Place Of Birth: * SAUDI ARABIA	Education Level: * High School
Number of Dependents: * 0	The purpose of opening the account: * For saving

Are you under the Foreign Account Tax Compliance Act (FATCA)?
Yes [✓ No]

Next

Figure16: Additional questions can be obtained from the Absher system


12:32

Landline Number:
0116363729

Relative or Friend Contact Information

First Name: *

Father Name: *

Grand Father Name: *

Family Name: *

Relationship: *

Mobile Number: *
e.g 966512345678

Landline Number:
e.g 0112046600

I?m the real beneficiary of the account
I hereby accept the Terms and Conditions

Figure17: Ask for friends' information on the registration page

4. Conclusion:

We found some security problems, which some developers may think it is simple, but it may become a Vulnerability that threatens the application, so attention must be paid to even those simple and small problems and solve it as soon as possible.

We also noticed a lack of interest in implementing clean code standards, including not using clear labels for variables, classes etc.

The exist of the application on Android open source systems means that all developers have the right to see and understand it, but due to the absence of documentation of the code and the lack of appropriate comments to clarify the used codes, made it difficult for us to understand many parts of the code, and also the absence of comments will make it difficult for future developers of the application.

And, from our perspective the use of design patterns would have been better if they had used FAÇADE and DECORATOR patterns ,for the reasons mentioned in the report.

We expect to resolve the issues mentioned in the report and improve the security and quality of the code to get better code and improve the usability of the application