# Alinma Bank App
## Quality Assessment

CS392: Software Engineering 2
1st Semester 1443H / Fall 2021

| Name | Email |
|---|---|
| Sadeem Faisal Alqahtani | sfaalqahtani29@sm.imamu.edu.sa |
| Sarah Khalid Alaradi | skmaloridi@sm.imamu.edu.sa |
| Shoug Ali Alsuhaibani | ssuhaibani@sm.imamu.edu.sa |
| Sarah Abdullah Alsarami | samalsarami@sm.imamu.edu.sa |

# Introduction

We outlined the studies and the quality assessments of Alinma Bank app in order to improve the overall quality of the app.

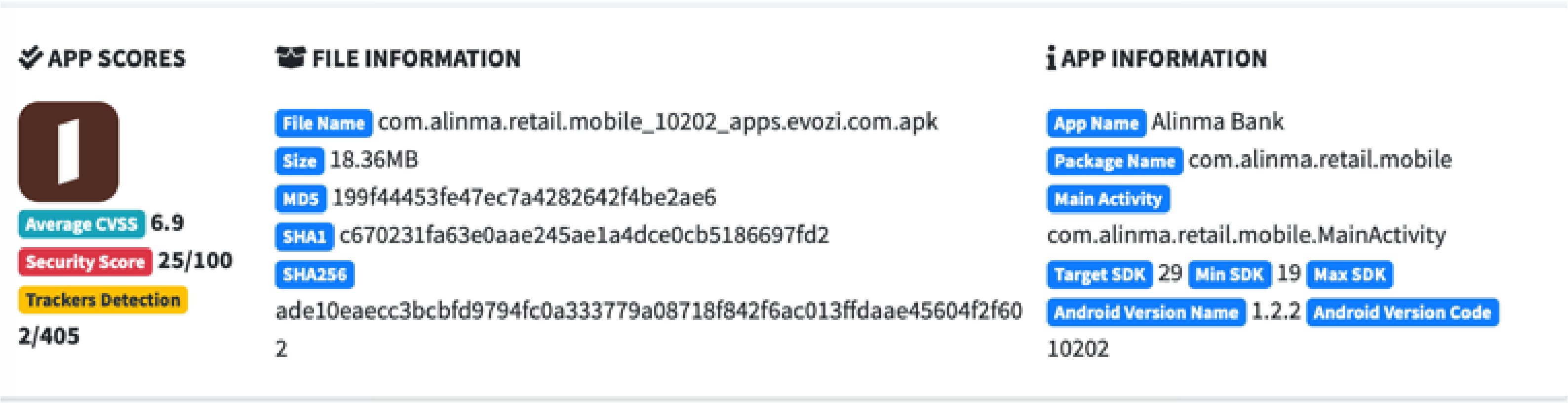
- This well be achieved by performing a static analysis on the Android version of the app by using the MobSF tool.
- Automated some of the source code quality analysis through the use of Codacy.

# Outline

- Alinma App Static Analysis.
- Security Issues.
- Clean Code Analysis.
- Code Documentation Analysis.

# Alinma App Static Analysis

The results given by the tool were: Security score, tracker detection, size of the file, latest app version and the different analysis options.



**APP SCORES**

Average CVSS 6.9

Security Score 25/100

Trackers Detection 2/405

**FILE INFORMATION**

File Name com.alinma.retail.mobile_10202_apps.evozi.com.apk

Size 18.36MB

MD5 199f44453fe47ec7a4282642f4be2ae6

SHA1 c670231fa63e0aae245ae1a4dce0cb5186697fd2

SHA256 ade10eaecc3bcbfd9794fc0a333779a08718f842f6ac013ffdaae45604f2f602

**APP INFORMATION**

App Name Alinma Bank

Package Name com.alinma.retail.mobile

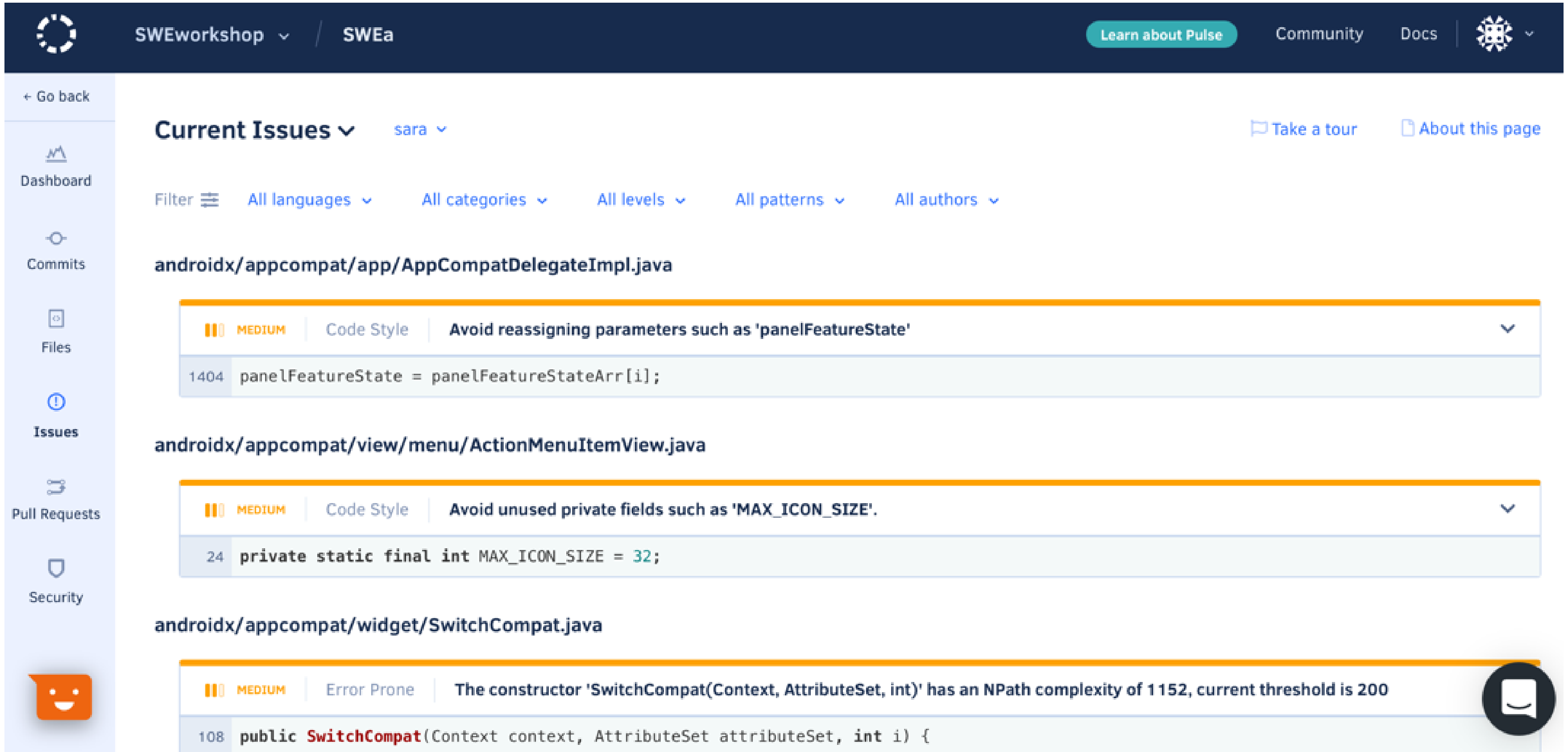
Main Activity com.alinma.retail.mobile.MainActivity

Target SDK 29 Min SDK 19 Max SDK

Android Version Name 1.2.2 Android Version Code 10202

An app score was shown, and the security score isn't very promising.

For Codacy, repository of java code was uploaded into Github. Then it was linked to Codacy and the issues for the code were available for preview.



androidx/appcompat/app/AppCompatDelegateImpl.java

MEDIUM | Code Style | Avoid reassigning parameters such as 'panelFeatureState'

```
panelFeatureState = panelFeatureStateArr[i];
```

androidx/appcompat/view/menu/ActionMenuItemView.java

MEDIUM | Code Style | Avoid unused private fields such as 'MAX_ICON_SIZE'.

```
private static final int MAX_ICON_SIZE = 32;
```

androidx/appcompat/widget/SwitchCompat.java

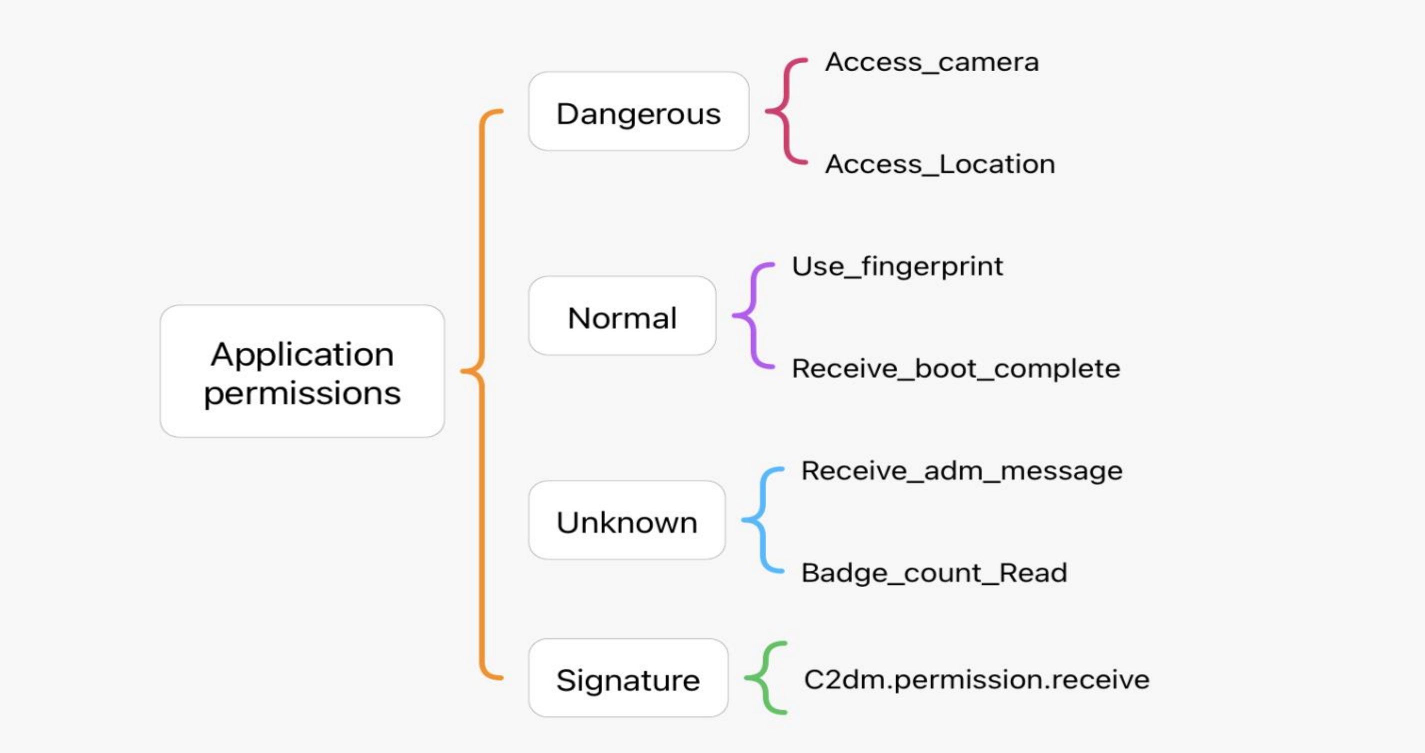
MEDIUM | Error Prone | The constructor 'SwitchCompat(Context, AttributeSet, int)' has an NPath complexity of 1152, current threshold is 200

```
public SwitchCompat(Context context, AttributeSet attributeSet, int i) {
```

# Security Issues

# Application Permissions

Type of application permissions and some examples of it:



# Manifest Analysis

- Avoid the Use of Clear Text traffic.
- Set the flag allowBackup to false.
- Protect the Broadcast Receiver.
- Protect the services from the presence of intent-filter.
- Protect the Content Provider.

# Code Analysis

- Don't make the app log sensitive data!
- Make sure to use an updated algorithm that is more secure.
- Use Internal storage instead of external to keep sensitive data hidden from users.
- Sensitive information should be encrypted and written to the database and make sure that input validation.
- Be aware sensitive data should not be copied to clipboard.

# Clean Code Analysis

# Meaningful Names

Names are everywhere in software, classes, functions and variables. So, they have a great importance in the continuity of the code.

- Has been used random names in class name such “zabt” ?.
- Many classes are named with one letter, so what does that mean? why this letter chosen?.
- The method name must be verb or verb phrases , when method name is “error” it is just word does not indicate verb, but it can be prefixed with is , get ,set to become a verb.
- You should avoid putting the name of a class as the same name of a method in it.
- You should not put a very long name for a method and put every name from the problem’s domain. it is too much.

# Functions

Functions are the first line of organization in any program. Writing them well save a lot of trouble.

- Functions should be Small, fulfilling the high Cohesion design principle.
- Avoid handling errors in the function.
- Avoid Duplication Blocks.
- Avoid Empty if Statement and switch Blocks.
- Listing numerous parameters better be Avoided.
- Give the method a descriptive name describing its functionality.

# Error Handling

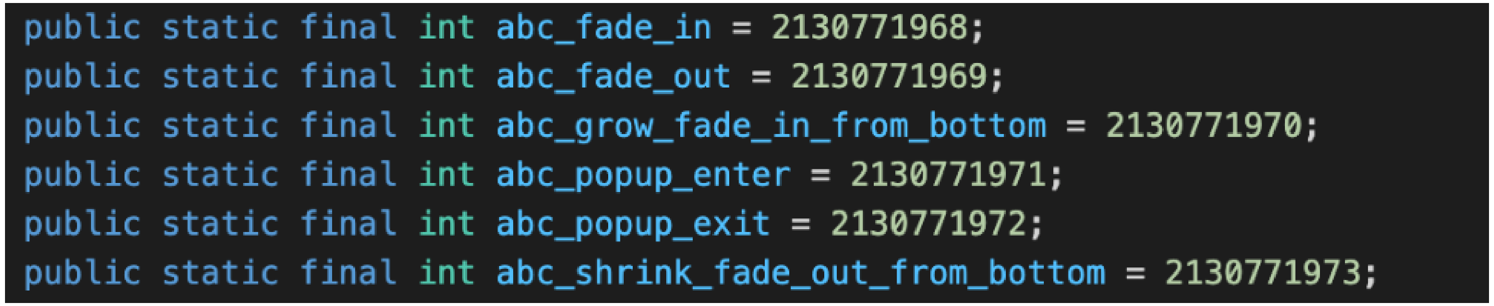
A programmer's responsibility is to ensure that his code does what it is supposed to do when error occurs.

- Instead of using an error flag or returning an error code, throw an exception when you encounter an error.
- Try writing tests that force exceptions and then adding behavior to your handler.
- Throwing a specific error/exception rather than a generic/general exception is recommended.
- Create informative error messages and pass them along with your exceptions. Mention the operation that failed and the type of failure. such as logging in your application.
- Avoid throwing NullPointerExceptions manually and throw an IllegalArgumentException instead.

# Formatting

Following a specific formatting method can really help the programmers when reading the code after sharing it with others, the code is then easier on the eyes.

- Proper indentation was followed through out the code.
- How ever vertical alignment wasn't followed in the code.



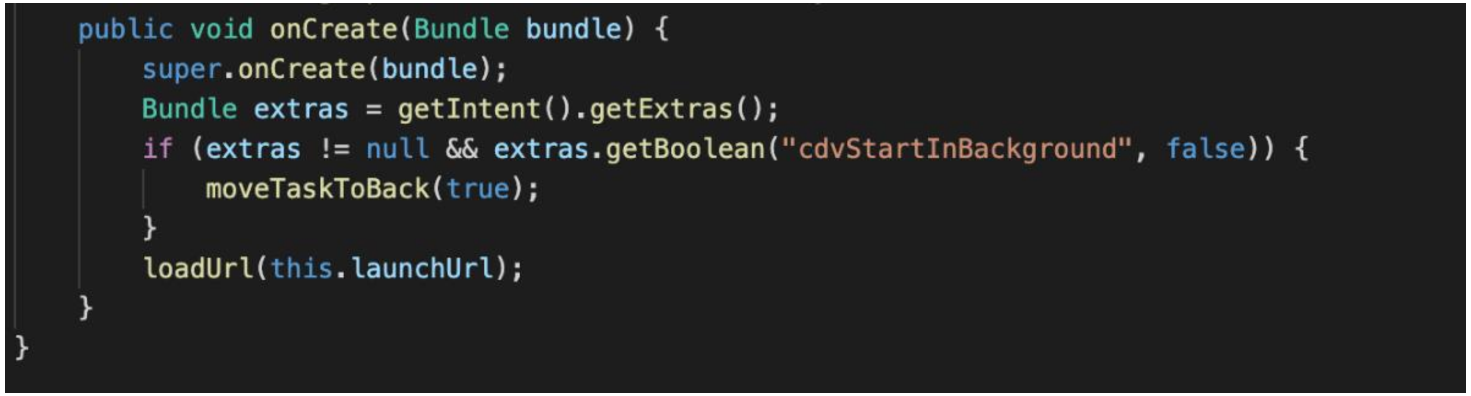
```
public static final int abc_fade_in = 2130771968;
public static final int abc_fade_out = 2130771969;
public static final int abc_grow_fade_in_from_bottom = 2130771970;
public static final int abc_popup_enter = 2130771971;
public static final int abc_popup_exit = 2130771972;
public static final int abc_shrink_fade_out_from_bottom = 2130771973;
```

# Clean Documentation Analysis

# Documentation

Documentation should compensate for the times where we cannot express the code and where its not clear. We should include parameter info in the comments, or where the return value is used for so other programmers could understand.

- Very poor documentation.
- Could be a security precaution.



```
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        Bundle extras = getIntent().getExtras();
        if (extras != null && extras.getBoolean("cdvStartInBackground", false)) {
            moveTaskToBack(true);
        }
        loadUrl(this.launchUrl);
    }
}
```

# Thank You for Your Attention!

## Any Question?

**You can see the full Documentation and Source code in Github**

https://github.com/CS392Workshop.git