



SOFTWARE ENGINEERING II

PROJECT - SECOND EVALUATION

Lecturer: Dr. Carlos Mera Gómez

Team 5 - ClientManagerApp

Members:

Daniel Garcia

David Neira

Jorge Pinargote

Milton García Cox

Guayaquil - Ecuador

2020 1S – P01

INDEX

1. Introduction	2
2. Evidence of adherence to SCRUM	2
Roles	2
Backlog	3
3. Architecture	4
4. Acceptance Testing	4
5. Use of a Build Automation Tool for the Project	6
6. Coding Standard Documentation	8
7. Detecting and Enforcing Coding Standards	8
8. Preemptive Error Detection	10
9. Adequate Use of a Teamwork Management Tool	11
10. Use of Continuous Integration Tool	12
11. Application Profiling	14
Flutter Performance in mode profile	14
DevTools	15
Performance overlay in main screen	16
12. Flow of windows and layout and System Deployment Guide	17
13. Use of an Authentication and Access-Control Framework	21
Authentication when the device has biometric type fingerprint	22
Authentication in device without biometric type fingerprint	22
14. Single Sign on	23
15. Pre-recorded video	23
16. Client acceptance	24
17. References	25

1. Introduction

This document provides a compilation of the generated documentation and the evidence corresponding to the work that has been carried out during all the Sprints of the application "ClientManagerApp".

The mobile application "ClientManagerApp" of the martial arts academy "8 Armas Team" allows the registration of clients and the monthly cash payments made to the academy. Customer data and payments can be saved for future consultation and management by the administration.

Additionally the source code and resources can be found in the following public Github repository: <https://github.com/demonpo/ClientManagerApp>.

2. Evidence of adherence to SCRUM

Roles

The roles and specifications for the members of the working group are detailed below.

Product Owner:

The person in charge of managing the flow of product value through the Product Backlog is Daniel Garcia. He works as an interlocutor with the stakeholder and as well has a role as a speaker of the requests and requirements of our client.

Scrum Master:

The one who is responsible for managing and ensuring that the Scrum process is carried out correctly is David Neira. He helps to eliminate the impediments that arise in the organization and that affect its ability to deliver value, and the integrity of this Scrum methodology.

Development Team:

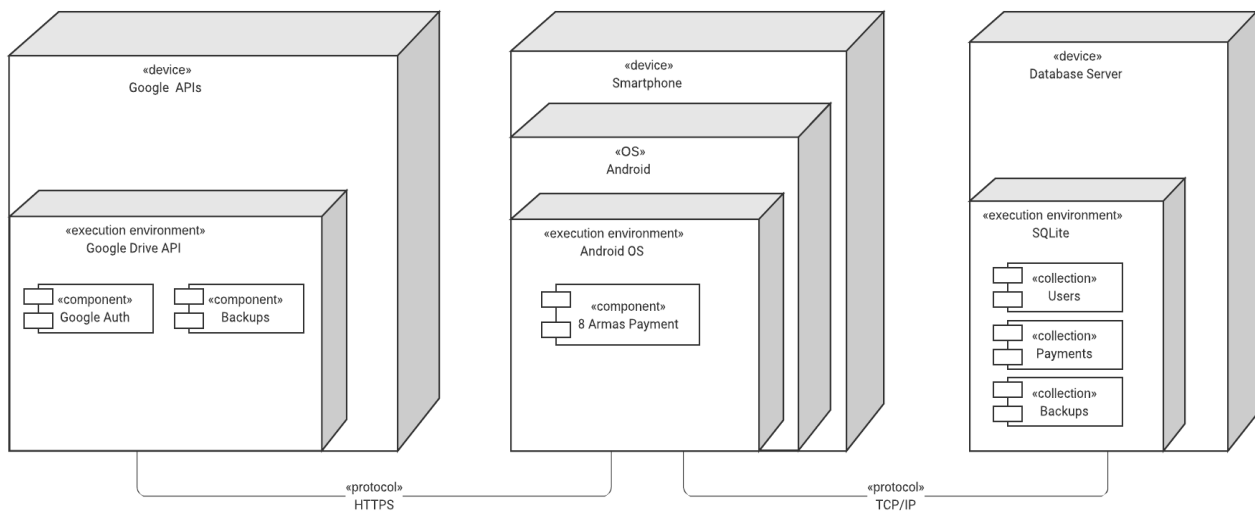
With the help of the members above, this group is also conformed by Milton Garcia Cox and Jorge Pinargote. They are in charge of developing the product, self-organizing and self-managing to deliver a version of the product.

Backlog

Scrum Product Backlog

User story	Story point(s)	Priority
As a user, I am able to create a client	2	1
As a user, I am able to eliminate a client	1	2
As a user, I am able to see the client list	1	3
As a user, I am able to edit a client	2	3
As a user, I am able to backup my data	3	4
As a user, I am able to get notification about payments delays	2	5
As a user, I am able to manage client payments	1	7
As a user, I can change my cellphone and also get all the data of the app	3	8
As a user, I am able save picture of my clients	3	9
As a user, I am able to authenticate via pin	4	10

3. Architecture



The 3-tier architecture designed for the ClientManagerApp application is detailed below.

We have the client's mobile device with Android OS where our application will be running.

Our app communicates with the SQLite embedded database server to store and directly obtain the information from the different tables and backups. The most important for us are the detailed users accounts and the detailed payments.

Additionally, the Google APIs from Drive are consumed for the authentication, storage and download service of backups of the Drive cloud database and the use of firebase for the temporary storage of information on the device.

4. Acceptance Testing



We have chosen Cucumber to write the acceptance tests, it uses gherkin, a natural language analyzer to guarantee the clarity of the writing of the scenarios that are specified in each feature to be tested. With regard to flutter, we've found that cucumber is the best-known available tool for writing acceptance tests. We've looked for optional libraries such as celery or testNG but they don't exist, because flutter is relatively new and it's normal that it doesn't have many alternatives in certain areas.

To use cucumber and gherkin in flutter it's necessary to write a name.feature, then inside, specify the scenario using the "given", "when" and "then" step structure, this would be enough to have all the acceptance cases ready and with the proper format to be able to present the user and keep effective contact between the end user and the tester.

```
1 Feature: Test del estado inicial de la app
2   Scenario: Cuando Se inicia la aplicación, existe un boton de ingreso.
3   Given estoy en el login
4   When doy tap a "Ingresar"
5   Then puedo ver la pantalla de home
```

Test Driver

Flutter provides a Gui tester called "Test Driver" that is based on Selenium, one of the best Gui testers on the market. With the help of this tool, we can emulate a lot of actions of the end user, such as a click, scroll or swipe. Currently, this is the flutter official library for integration tests or acceptance tests.

Cucumber and Test Driver connection

Another reason because we chose cucumber is that it allows us to integrate easily with the flutter test driver, so we could actually execute each of the acceptance tests written above, launching each event that the end user could perform but in a programmatic way.

It's only necessary to write each of the steps that are part of a scenario in a step.dart, each step written in cucumber can send the variables that are considered necessary for its implementation,

in this way we achieve that the end user has the possibility of understand and change even the flow of execution from a more understandable language without need to deploy each of the steps.

```
import 'package:flutter_driver/flutter_driver.dart';
import 'package:flutter_gherkin/flutter_gherkin.dart';
import 'package:gherkin/gherkin.dart';

class EstoyHome extends ThenWithWorld<FlutterWorld> {
  EstoyHome()
    : super(StepDefinitionConfiguration()..timeout = Duration(seconds: 10));

  @override
  Future<void> executeStep() async {
    final home = find.byValueKey("HomePage");
    expect(await FlutterDriverUtils.isPresent(world.driver, home), true);
  }

  @override
  RegExp get pattern => RegExp(r"puedo ver la pantalla de home");
}
```

5. Use of a Build Automation Tool for the Project

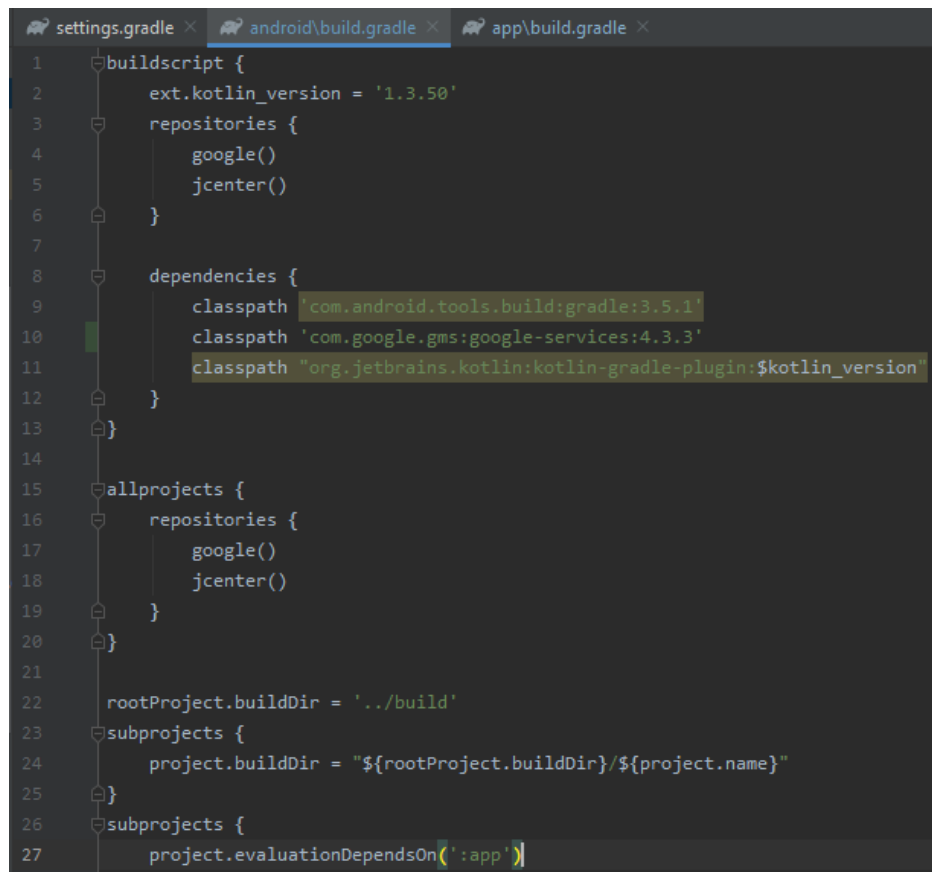
We have used Gradle for build automation in our application.



Gradle is an open-source build automation system that builds upon the concepts of Apache Ant and Apache Maven and introduces a Groovy-based domain-specific language instead of the XML form used by Maven for declaring the project configuration. Gradle uses a directed acyclic graph to determine the order in which tasks can be run.

Among the benefits that Gradle provides us over Maven are that it is the official build system for Android and has support for various technologies, languages and frameworks including Flutter.

Gradle is very flexible and customizable, therefore compilations are easier to do and allows incremental construction since in the task compilation process it validates if the input or output has changed. If there is no change, the consideration is updated, and it is not executed. It means that, in most cases, it speeds up the average compilation time.



```
1  buildscript {
2      ext.kotlin_version = '1.3.50'
3      repositories {
4          google()
5          jcenter()
6      }
7
8      dependencies {
9          classpath 'com.android.tools.build:gradle:3.5.1'
10         classpath 'com.google.gms:google-services:4.3.3'
11         classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"
12     }
13 }
14
15 allprojects {
16     repositories {
17         google()
18         jcenter()
19     }
20 }
21
22 rootProject.buildDir = '../build'
23 subprojects {
24     project.buildDir = "${rootProject.buildDir}/${project.name}"
25 }
26 subprojects {
27     project.evaluationDependsOn(':app')
```



```
android {
    compileSdkVersion 28

    sourceSets {
        main.java.srcDirs += 'src/main/kotlin'
    }

    lintOptions {
        disable 'InvalidPackage'
    }

    defaultConfig {
        // TODO: Specify your own unique Application
        applicationId "com.demonpo.clientmanagerapp"
        minSdkVersion 18
        targetSdkVersion 28
        versionCode flutterVersionCode.toInteger()
        versionName flutterVersionName
    }
}
```


6. Coding Standard Documentation

Reference of the coding standards for Flutter and Dart can be found in the following links:

<https://github.com/flutter/flutter/wiki/Style-guide-for-Flutter-repo#philosophy>

<https://dart.dev/guides/language/effective-dart/style>

This project is following the standards mentioned before because all the development is based on a Flutter environment.

We have used the style guides for Dart code proposed by the Flutter community. It includes the philosophy for writing code and documentation. It also includes features to run and generate tests. We chose this because it is well documented and detailed. It truly represents a standard for the Flutter developer community.

7. Detecting and Enforcing Coding Standards

The project is using pedantic library to enforce coding standards:

<https://pub.dev/packages/pedantic>

The static analysis provided by Pedantic shows us the best practices for the code we write. The other option was Effective Dart but we chose Pedantic because the lints are stricter for the sake of consistency and also pedantic is used by google team when they are developing with dart programming language.

Flutter framework provides checkstyle via “flutter analyze” command, with this command you can now in what lines of code there are style errors.

This project is using the following rules (same as google company):

- `always_declare_return_types`
- `always_require_non_null_named_parameters`
- `annotate_overrides`
- `avoid_empty_else`
- `avoid_init_to_null`

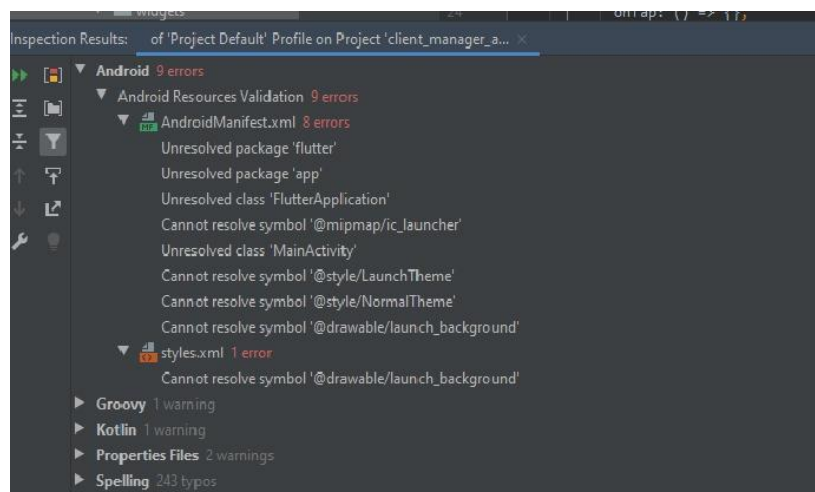
- `avoid_null_checks_in_equality_operators`
- `avoid_relative_lib_imports`
- `avoid_return_types_on_setters`
- `avoid_shadowing_type_parameters`
- `avoid_types_as_parameter_names`
- `camel_case_extensions`
- `curly_braces_in_flow_control_structures`
- `empty_catches`
- `empty_constructor_bodies`
- `library_names`
- `library_prefixes`
- `no_duplicate_case_values`
- `null_closures`
- `omit_local_variable_types`
- `prefer_adjacent_string_concatenation`
- `prefer_collection_literals`
- `prefer_conditional_assignment`
- `prefer_contains`
- `prefer_equal_for_default_values`
- `prefer_final_fields`
- `prefer_for_elements_to_map_fromIterable`
- `prefer_generic_function_type_aliases`
- `prefer_if_null_operators`
- `prefer_is_empty`
- `prefer_is_not_empty`
- `prefer_iterable_whereType`
- `prefer_double_quotes`
- `prefer_spread_collections`
- `recursive_getters`
- `slash_for_doc_comments`
- `type_init_formals`
- `unawaited_futures`
- `unnecessary_const`
- `unnecessary_new`

- unnecessary_null_in_if_null_operators
- unnecessary_this
- unrelated_type_equality_checks
- use_function_type_syntax_for_parameters
- use_rethrow_when_possible
- valid_regexp

8. Preemptive Error Detection

We use Android Studio for the development of the application, so our project has been supported by IDE's own code analysis tools. The Code Cleanup feature has been used and Lint checks for inspect code.

The Code Cleanup of Android Studio starts inspecting code defined rules. The Lint tool helps find unstructured code that can affect the reliability and efficiency of our project.



These tools can be applied at the project compilation or manually from the IDE plugins.

DartAnalyzer

The dartanalyzer command performs the same static analysis that you get when you use an IDE or editor that has Dart support. Dartanalyzer applies more restrictive rules to the type system

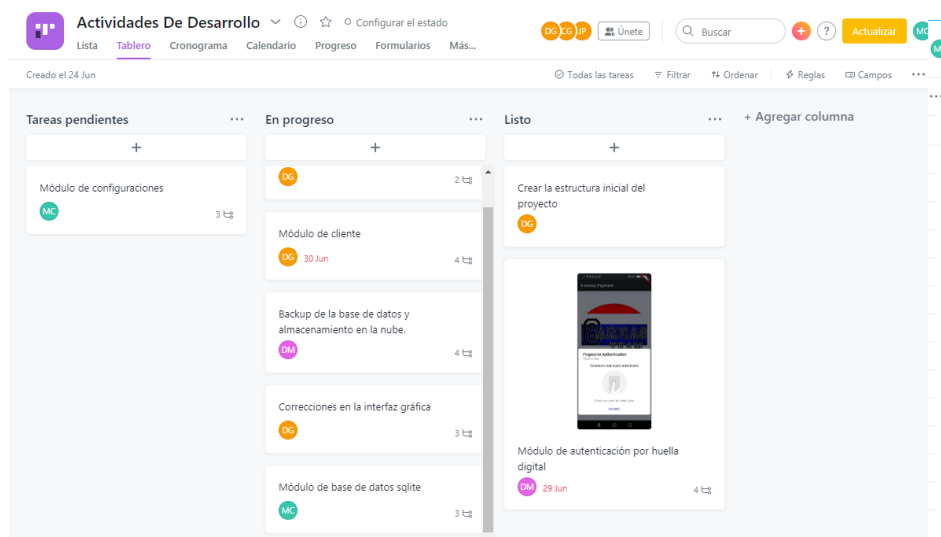
and, as a result, finds more errors during static analysis and at runtime. Another benefit is a faster compilation.

```
C:\Users\david\Desktop\ClientManagerApp>dartanalyzer lib
Analyzing lib...
lint - The method abonos should have a return type but doesn't. - lib\Abono\bloc\abono_bloc.dart:21:7 - always_declare_return_types
lint - The method getAbonos should have a return type but doesn't. - lib\Abono\bloc\abono_bloc.dart:27:3 - always_declare_return_types
lint - The method addAbono should have a return type but doesn't. - lib\Abono\bloc\abono_bloc.dart:45:3 - always_declare_return_types
lint - The method updateAbono should have a return type but doesn't. - lib\Abono\bloc\abono_bloc.dart:50:3 - always_declare_return_types
lint - The method deleteAbonoById should have a return type but doesn't. - lib\Abono\bloc\abono_bloc.dart:55:3 - always_declare_return_types
lint - 'Future' results in 'async' function bodies must be 'await'ed or marked 'unawaited' using 'package:pedantic'. - lib\Abono\bloc\abono_b
lint - The method deleteAllAbonosByClientId should have a return type but doesn't. - lib\Abono\bloc\abono_bloc.dart:60:3 - always_declare_ret
lint - Annotate overridden members. - lib\Abono\bloc\abono_bloc.dart:65:3 - annotate_overrides
lint - The method dispose should have a return type but doesn't. - lib\Abono\bloc\abono_bloc.dart:65:3 - always_declare_return_types
```

9. Adequate Use of a Teamwork Management Tool

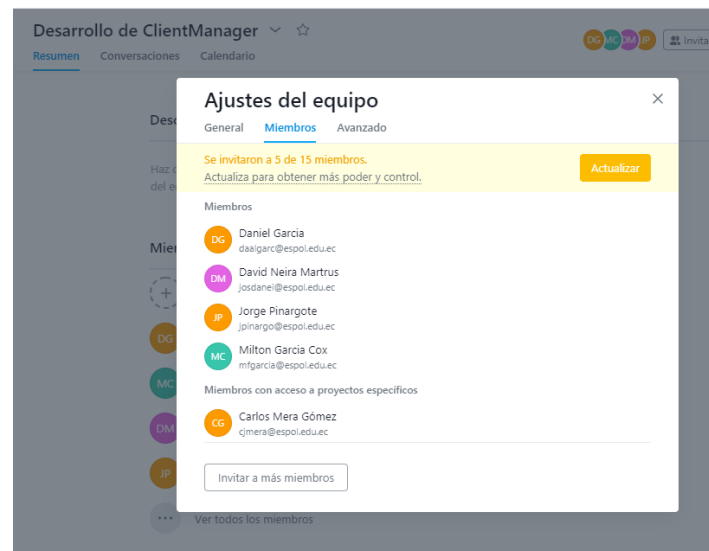
Asana

We have chosen Asana for teamwork management, other options were clickup and trello, but Asana was the winner because it has a better user interface in our opinion. It's a web and mobile application designed to help teams organize, track, and manage their work.



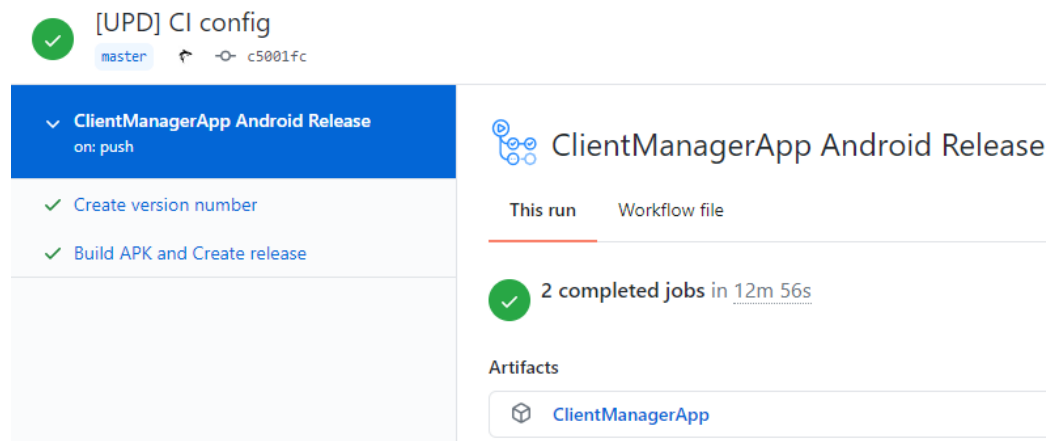
We have structured our ClientManager Development team in Asana so that the tasks can be divided into 3 categories: “Pendientes”, “En Progreso” y “Listo”.

Our calendar is made up of main tasks made up of subtasks with small time intervals




10. Use of Continuous Integration Tool

To perform the continuous integration, github actions were used, a flutter workflow was created with a .yaml configuration file that describes all the actions that are carried out to deploy the system and generate the deliverables in .apk format.



Configuration file with actions

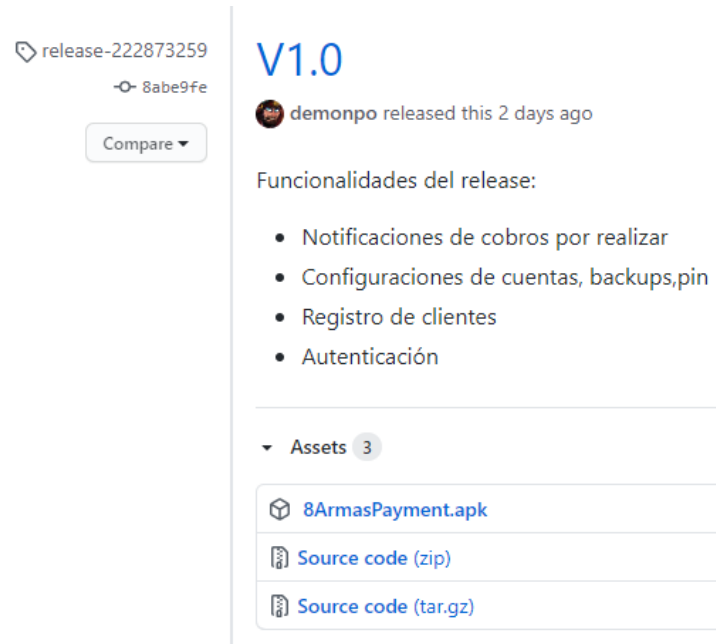
```
# Upload generated apk to the artifacts.
- uses: actions/upload-artifact@v1
  with:
    name: ClientManagerApp
    path: build/app/outputs/apk/release/app-release.apk
- name: Create Release
  id: create_release
  uses: actions/create-release@latest
  env:
    GITHUB_TOKEN: ${secrets.GH_TOKEN}
  with:
    tag_name: release-${github.run_id}
    release_name: Release-${github.run_id}
    body: |
      Funcionalidades del release:
      - Notificaciones de cobros por realizar
      - Configuraciones de cuentas, backups, pin
      - Registro de clientes
      - Autenticación
    draft: false
    prerelease: false
```

 [UPD] CI config

master ↶ ↷ c5001fc

✓ ClientManagerApp Android Release on: push	ClientManagerApp Android Release / Build APK and Create release succeeded 17 hours ago in 12m 8s
✓ Create version number	
✓ Build APK and Create release	<ul style="list-style-type: none">▶ ✓ Set up job▶ ✓ Run actions/checkout@v1▶ ✓ Run actions/setup-java@v1▶ ✓ Run subosito/flutter-action@v1▶ ✓ Run flutter pub get▶ ✓ Run flutter pub run import_sorter:main▶ ✓ Run flutter format lib▶ ✓ Run flutter build apk▶ ✓ Run actions/upload-artifact@v1▶ ✓ Create Release▶ ✓ Post Run actions/setup-java@v1▶ ✓ Complete job

Release of the application in format APK

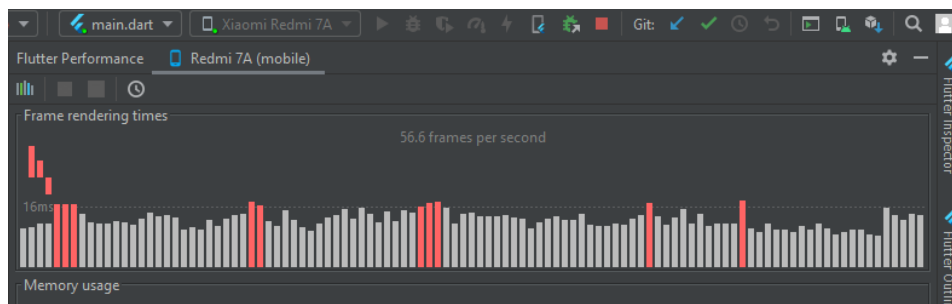


11. Application Profiling

In Android Studio Flutter's profile mode compiles and launches the application almost identically to release mode, but with just enough additional functionality to allow debugging performance problems.

In profile mode, some debugging ability is maintained enough to profile app's performance. Profile mode is disabled on the emulator and simulator because their behavior is not representative of real performance.

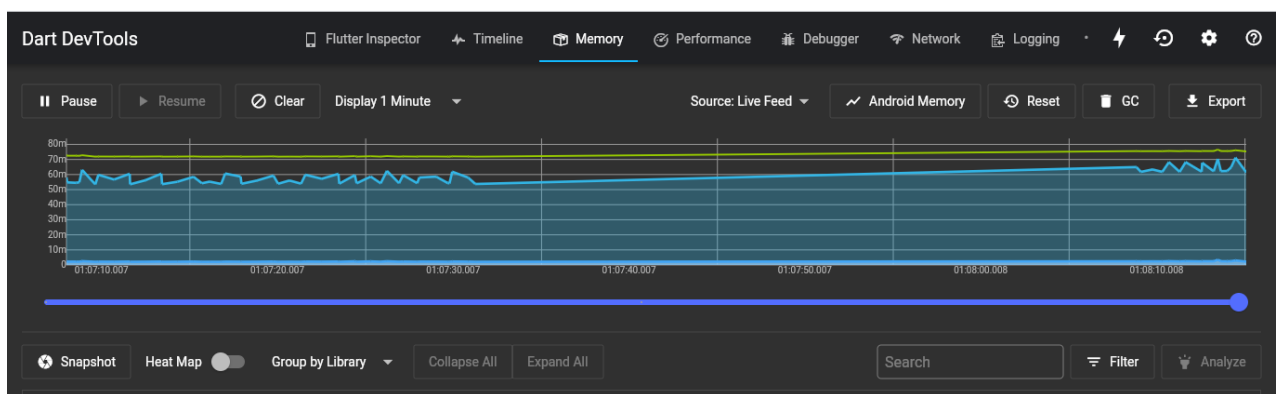
Flutter Performance in mode profile



When the app is running in profile mode, we launch DevTools.

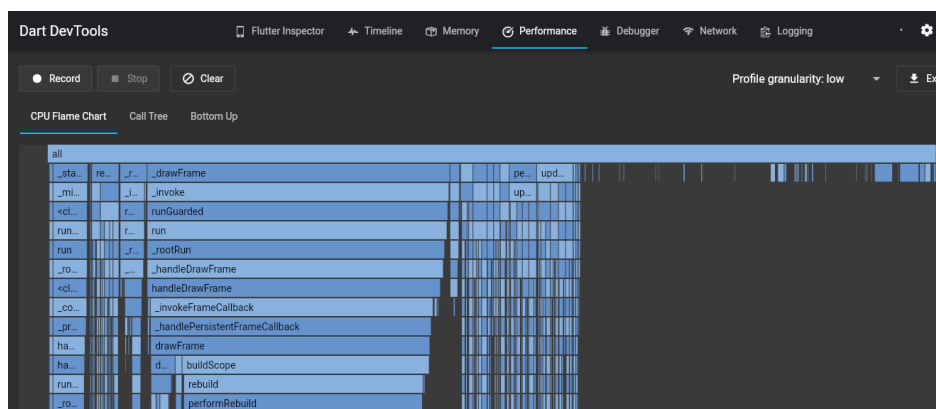
DevTools

DevTools provides features like profiling, examining the heap, displaying code coverage, enabling the performance overlay, and a step-by-step debugger. DevTools' Timeline view allows to investigate the UI performance of us application on a frame-by-frame basis.



CPU Flame Chart

This tab of the profiler shows CPU samples for the recorded duration. This chart should be viewed as a top-down stack trace, where the top-most stack frame calls the one below it. The width of each stack frame represents the amount of time it consumed the CPU. Stack frames that consume a lot of CPU time may be a good place to look for possible performance improvements.

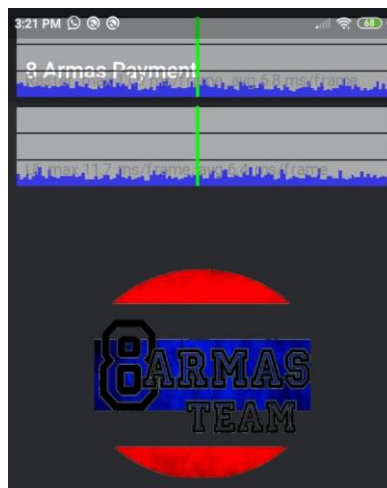


Performance overlay in main screen

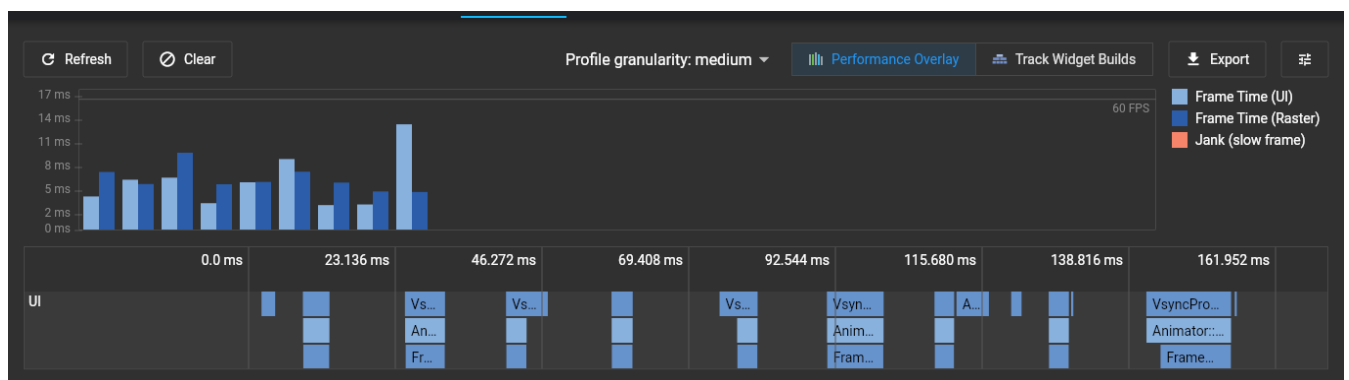
The performance overlay displays statistics in two graphs that show where time is being spent in the app. If the UI is janky (skipping frames), these graphs help you figure out why. The graphs display on top of the running app, but they are not drawn like a normal widget—the Flutter engine itself paints the overlay and only minimally impacts performance. Each graph represents the last 300 frames for that thread.

The following screenshot shows the performance overlay running on the Flutter App that is being develop:

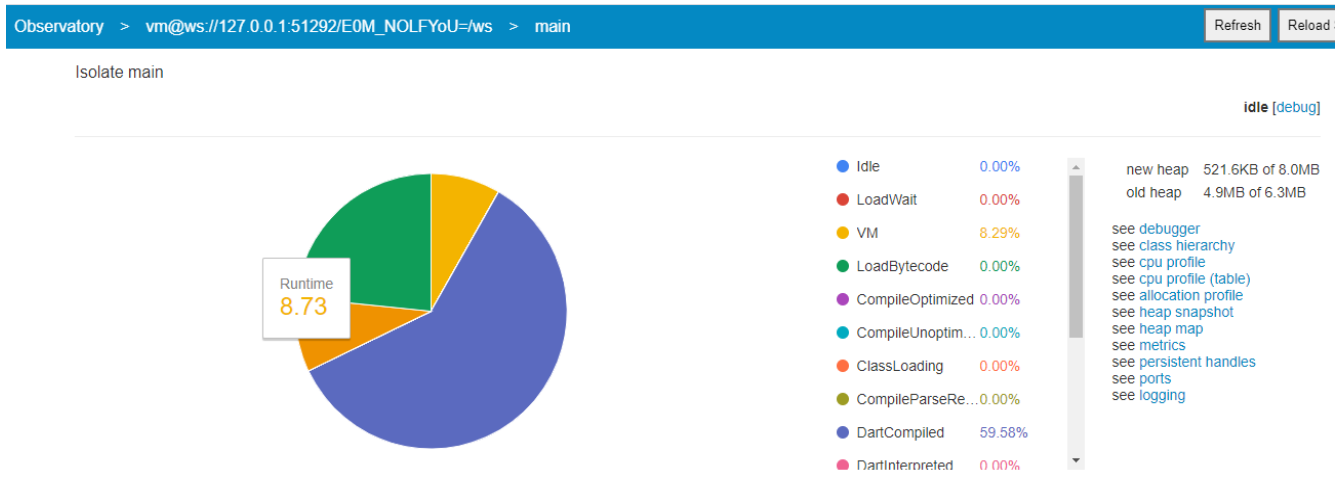
Performance Overlay in SmartPhone with Android



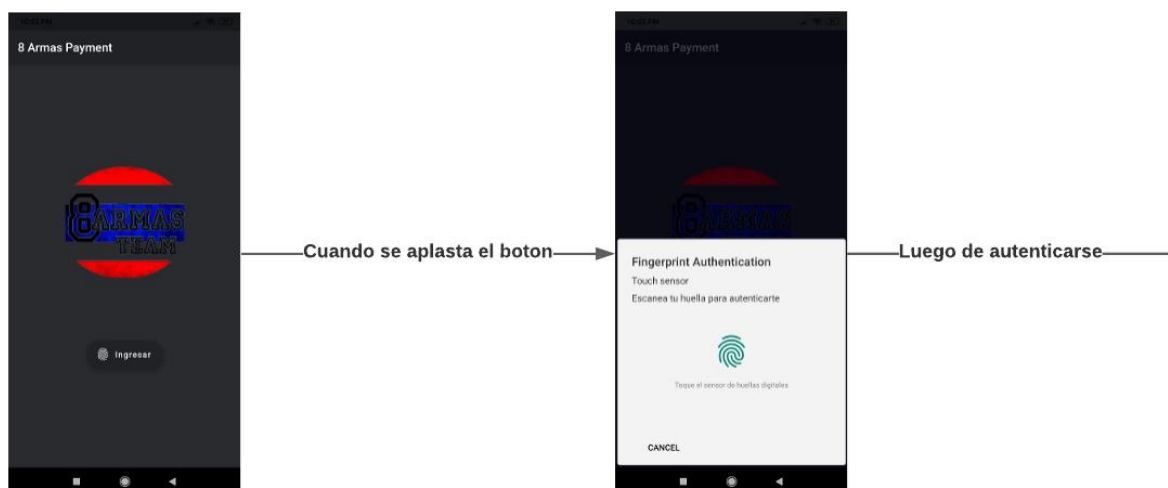
Performance Overlay in DevTools

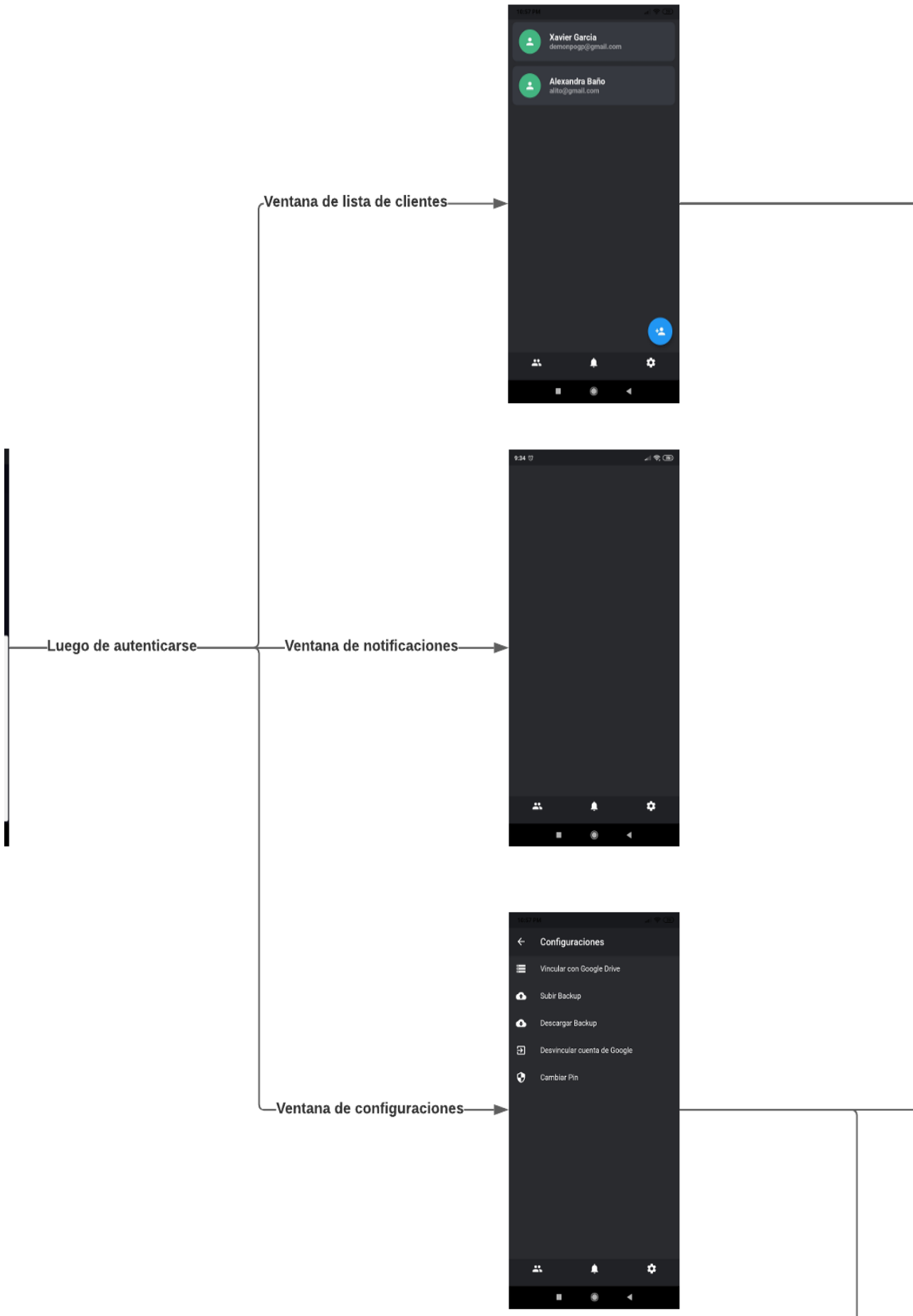


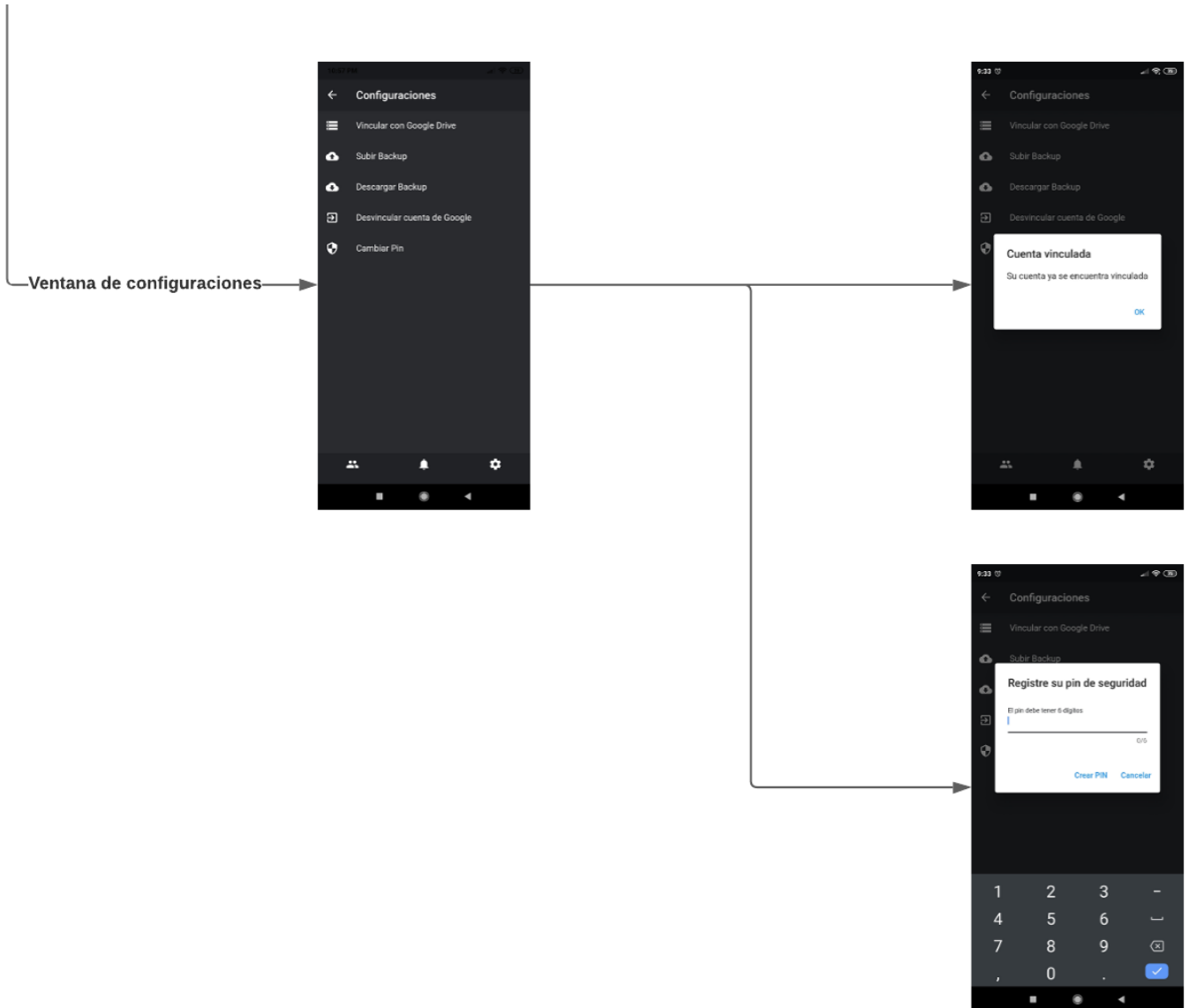
An Observatory debugger and profiler



12. Flow of windows and layout and System Deployment Guide

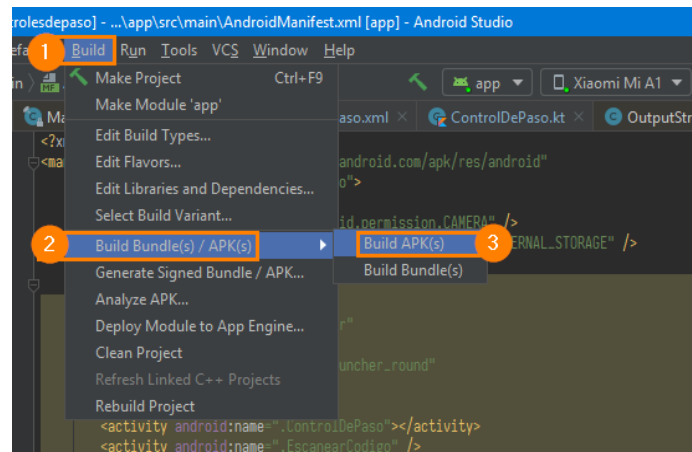








With Android Studio it is possible to export our project to an APK file by a previous compilation. It is released through the emulator or directly on an Android smartphone.



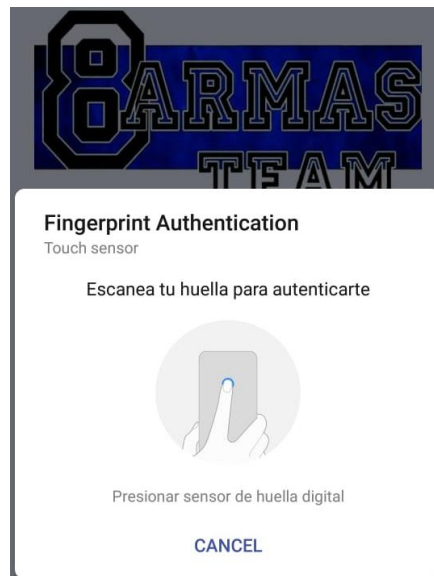
Our Android executable is a file with the .APK extension. This format is a variant of the Java JAR format and is used to distribute and install packaged components for the Android platform. The client needs to accept apks from unknown sources in his phone settings.

13. Use of an Authentication and Access-Control Framework

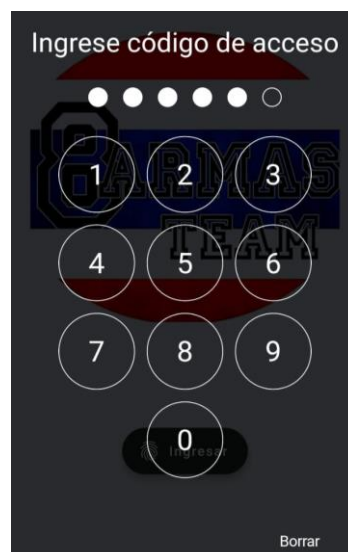
For user access control, an authentication plugin called `local_auth` is used that is available in `pub.dev`. This Flutter plugin provides means to perform local authentication on the user's device, it uses biometric authentication on iOS (Touch ID or lock code) and fingerprint APIs on Android (introduced in Android 6.0).

Currently the biometric type fingerprint is implemented, and when the device does not have biometric sensors available, the authentication is done using a numerical access code.

Authentication when the device has biometric type fingerprint

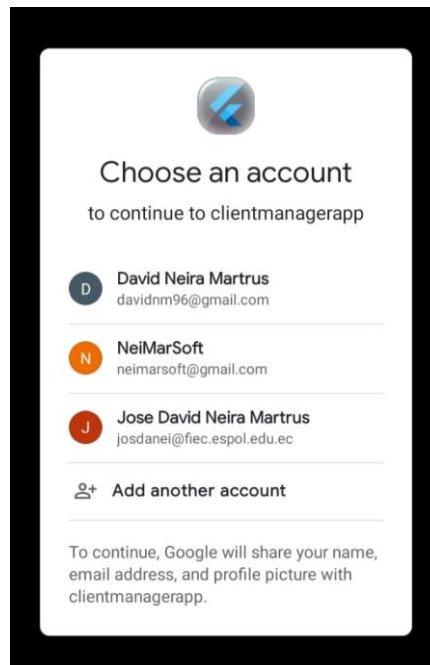


Authentication in device without biometric type fingerprint



14. Single Sign on

The Google SSO allows its users to log in only once to gain access to all their applications, in this case the user when logging in with his Google account in the settings section, will have access to Google Drive to allow him to store backups of the database.



15. Pre-recorded video

In the following link you can access the 20-minute video that is made available to the teacher, authorities and client.

Link: <https://drive.google.com/drive/folders/1otxMyfuGN5-IchqSVX6xhOqQm5O-VnE3?usp=sharing>

16. Client acceptance

GUAYAQUIL, 25 DE AGOSTO DEL 2020

CERTIFICADO DE ACEPTACIÓN

Yo, **Francisco Bajaña Malo** con cédula de ciudadanía # **0922004825**, certifico conocer y aceptar el trabajo realizado durante los Sprints del proyecto "8 Armas Payment" para el manejo de clientes y pagos en mi negocio.

En consecuencia autorizo a los señores: Daniel García, David Neira, Jorge Pinargote y Milton García Cox; miembros del grupo de trabajo de la aplicación para que hagan uso de esta certificación como mejor lo estimen conveniente.

espol

Atentamente.



Francisco Bajaña Malo
C.C.: 0922004825

17. References

- [1] <https://github.com/flutter/flutter/wiki/Style-guide-for-Flutter-repo#philosophy>
- [2] <https://flutter.dev/docs/perf/rendering/ui-performance>
- [3] <https://github.com/demonpo/ClientManagerApp>
- [4] <https://pub.dev/packages/pedantic>
- [5] <https://flutter-es.io/docs/deployment/android>
- [6] Essential Scrum: A Practical Guide to the Most Popular Agile Process, Kenneth S. Rubin
- [7] Software Engineering, Ian Sommerville