DigiProf Assurance Plan

Group 13

1/ Software Tools

- Automated unit testing tools/frameworks:

Junit 5	• For Java. • Simple • Test data is first tested and then inserted in the piece of code -> test-driven.	• Free	Already in Android Studio

- Integration testing tools/frameworks:

JUnit	• For Java. • Simple • Test data is first tested and then inserted in the piece of code -> test-driven.	• Free	Built-in in android studio

- Automated software testing tools for system testing and software testing in general:

Espresso	• In java and can be used to test Android apps. • No need to recompile app. • Allows parallel execution of test scripts. • A small change does not require re-installation of the application. • A bit slow compared to other apps. • Easy to use.	• Free	Built-in in Android Studio

Android Studio Manual Tests:

- Validation testing: ensure that the hardware/software system meets the requirements allocated to software as identified initially. This one is done manually.

- User acceptance testing: This part is for users to use the workable version of the app and test out features -> The testing is performed manually.

- Writing tests in parallel to the development of the task help enhance team collaboration. Test cases are written down and managed at all times in a doc file. Any modification to the app will result in the changing of some of the old cases.

2/ Internal deadlines:

Unit testing	11/2/20
Integration Testing	11/4/20
System Testing	11/6/20
Acceptance Testing	11/9/20

All of the testing will be performed frequently and regression testing is done by the end of every system testing.

3/ User acceptance testing:
The version 3 of the app is speculated to be usable by 12/07/2020. Therefore, the initial plan is to have the version 2 get tested on 11/16/2020 and finish within 2 days. The user acceptance testing for version 3 of the app will start on 11/23/2020 and be carried out for the following days.

First of all, the user acceptance testing scope is defined. This also includes all of the assumptions and constraints of the test. All of the test cases will be identified to make sure that they sufficiently cover most of User Acceptance Testing scenarios. The users will then be asked to use the app and provide feedback. After the testing is done, we will assess the collected data to improve future test cases and UAT workflows.

4/ Integration testing approach:
This is the approach we used: Sandwich/ Hybrid which is a combination of Top Down and Bottom Up approaches.
It is a mix of making the top-level units and lower level units at different stages in no particular order. First we implemented and tested the record of a video feature, then the upload video feature, then the login/registration/forgot password page and finally the display of all videos on the home-screen feature.

5/ Metrics collection tools:
We used the Code::Stats plug-in in Android Studio to generate all of the needed metrics including size and number of code lines. Microsoft Excel would then be used to create line graphs from the accumulated data.

Our Metrics For Version 1 of DigiProf (Table Form):

Java Class	Total Lines of Code	File Type	File Size (in kB)
AdapterVideo.Java	163	bat	2
AddVideoActivity.Java	306	gitignore	0
ExampleInstrumentedTest.Java	26	gradle	2
ExampleUnitTest.Java	17	java	37
ForgotPasssword.Java	95	json	1
MainActivity.Java	133	md	4
ModelVideo.Java	67	pdf	102
User.Java	27	pro	0
UserRegistration.Java	128	properties	1
VideoActivity.Java	98	xml	26
		Project Total	173

File Type	Number of Files
bat	1
gitignore	2
gradle	3
java	10
json	1
md	1
pdf	1
pro	1
properties	3
xml	19
Total number of files in project	42

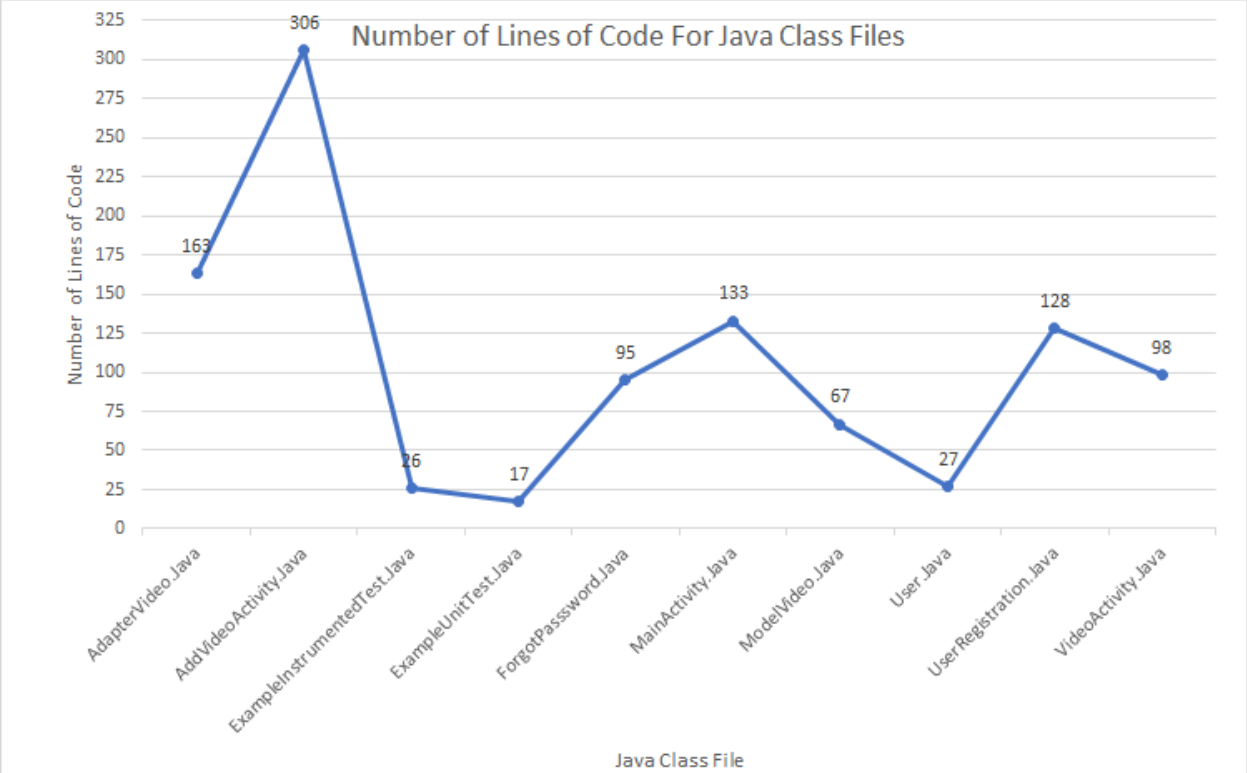
Properties File	Number of lines of Code

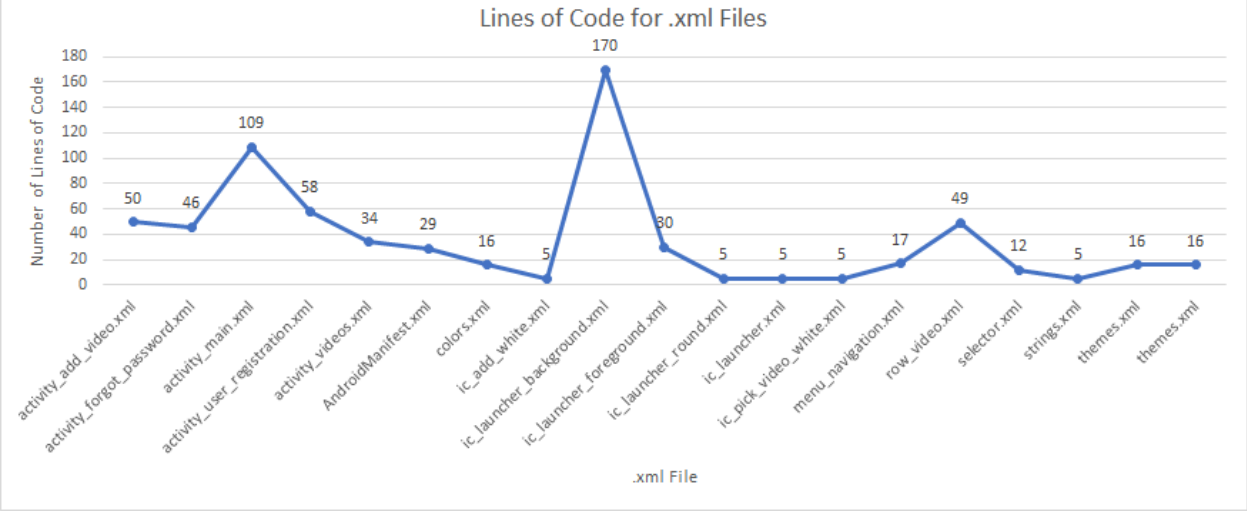
gradle.properties	19
gradle-wrapper.properties	6
local.properties	8

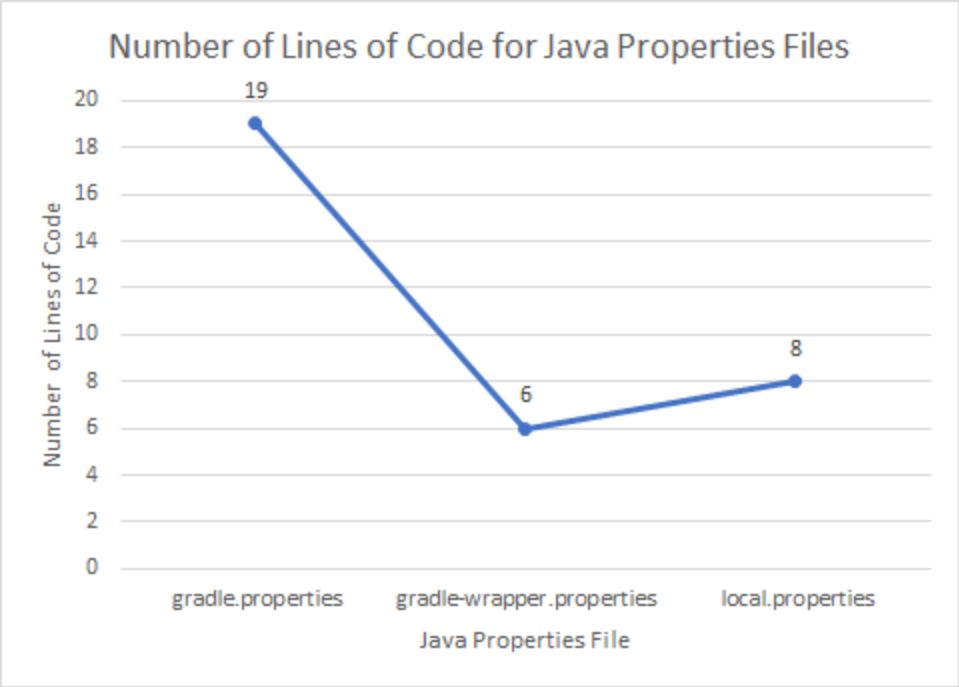
.XML File	Total Lines of code
activity_add_video.xml	50
activity_forgot_password.xml	46
activity_main.xml	109
activity_user_registration.xml	58
activity_videos.xml	34
AndroidManifest.xml	29
colors.xml	16
ic_add_white.xml	5
ic_launcher_background.xml	170
ic_launcher_foreground.xml	30
ic_launcher_round.xml	5

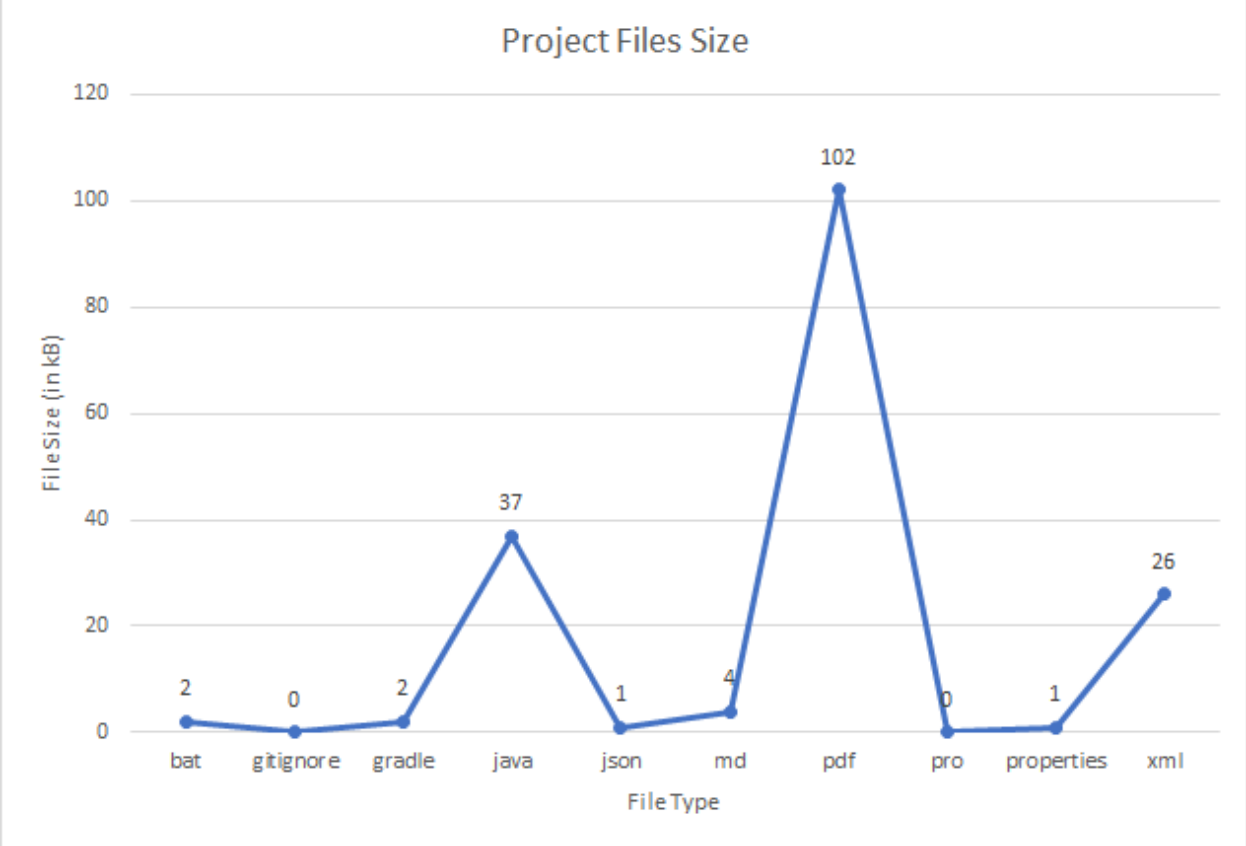
ic_launcher.xml	5
ic_pick_video_white.xml	5
menu_navigation.xml	17
row_video.xml	49
selector.xml	12
strings.xml	5
themes.xml	16
themes.xml	16

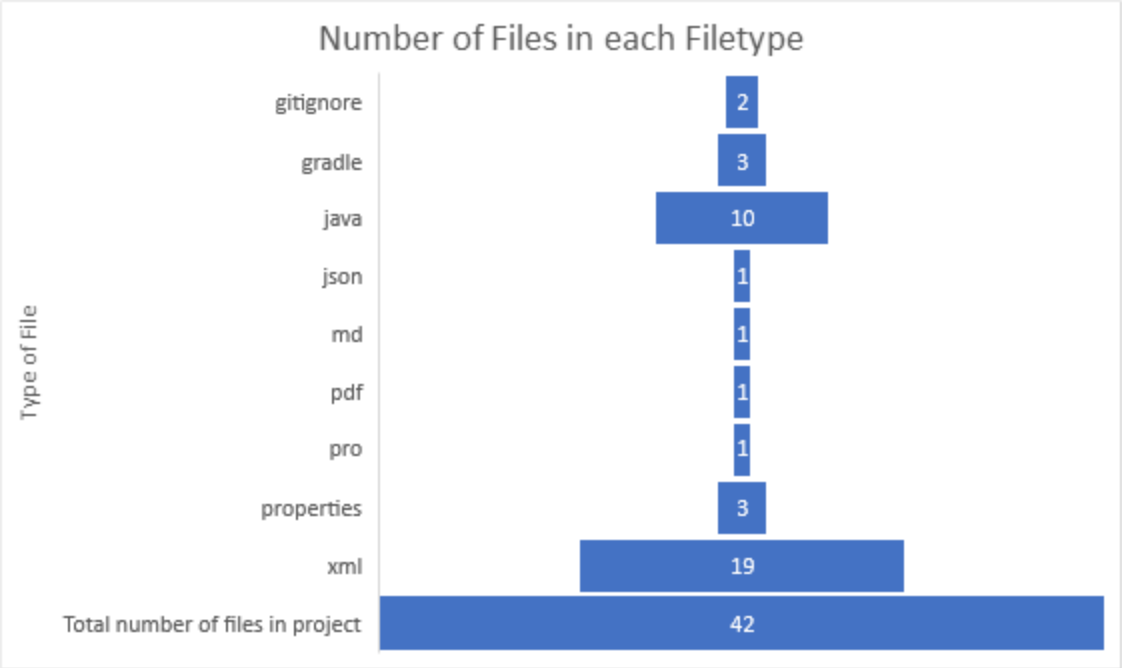
Our Metrics For Version 1 of DigiProf (Line Graph Form):


Number of Lines of Code For Java Class Files
Number of Lines of Code
325
300
275
250
225
200
175
150
125
100
75
50
25
0
163
306
26
17
95
133
67
27
128
98
AdapterVideo.Java
AddVideoActivity.Java
ExampleInstrumentedTest.Java
ExampleUnitTest.Java
ForgotPassword.Java
MainActivity.Java
ModelVideo.Java
User.Java
UserRegistration.Java
VideoActivity.Java
Java Class File


Lines of Code for .xml Files
Number of Lines of Code
180
160
140
120
100
80
60
40
20
0
50
46
109
58
34
29
16
5
170
30
5
5
5
17
49
12
5
16
16
activity_add_video.xml
activity_forgot_password.xml
activity_main.xml
activity_user_registration.xml
activity_videos.xml
AndroidManifest.xml
colors.xml
ic_add_white.xml
ic_launcher_background.xml
ic_launcher_foreground.xml
ic_launcher_round.xml
ic_launcher.xml
ic_pick_video_white.xml
menu_navigation.xml
row_video.xml
selector.xml
strings.xml
themes.xml
themes.xml
.xml File


Number of Lines of Code for Java Properties Files
Number of Lines of Code
20
18
16
14
12
10
8
6
4
2
0
19
6
8
gradle.properties
gradle-wrapper.properties
local.properties
Java Properties File


Project Files Size
File Size (in kB)
120
100
80
60
40
20
0
2
0
2
37
1
4
102
0
1
26
bat
gitignore
gradle
java
json
md
pdf
pro
properties
xml
File Type



6/ Additional actions

As a team, we take further steps to ensure the quality of the app:

- Pair programming is preferred among members because some of it appears to be an effective way to share knowledge and makes sure that fewer coding mistakes are made.
- Code review session is regularly held with Andy Wang – our Project manager being the host as he goes through the code and provides feedback. The session can be seen as a place for everyone to receive criticisms on their obscure code and improve themselves.
- We have a fixed stand-up meeting schedule every Monday. This helps avoid all the long, redundant and meaningless meetings that we would normally have at the beginning of the project. Furthermore, it allows all the team member to have more time to focus on their assigned tasks.

References:

[1] "20 Most Popular Unit Testing Tools in 2020", *Softwaretestinghelp.com*, 2020. [Online]. Available: https://www.softwaretestinghelp.com/unit-testing-tools/. [Accessed: 31- Oct- 2020].

[2] "CyVis - Software Complexity Visualiser", *Cyvis.sourceforge.net*, 2020. [Online]. Available: http://cyvis.sourceforge.net/. [Accessed: 31- Oct- 2020].

[3] "Integration Testing - SOFTWARE TESTING Fundamentals", *SOFTWARE TESTING Fundamentals*, 2020. [Online]. Available: https://softwaretestingfundamentals.com/integration-testing/. [Accessed: 31- Oct- 2020].

[4] "Top 5 Automated Testing Tools for Android: October 2018", *Bugfender.com*, 2020. [Online]. Available: https://bugfender.com/blog/best-automated-testing-tools-android/. [Accessed: 31- Oct- 2020].

[5] "Top 10 Integration Testing Tools to Write Integration Tests", *Softwaretestinghelp.com*, 2020. [Online]. Available: https://www.softwaretestinghelp.com/integration-testing-tools/. [Accessed: 31- Oct- 2020].