



Faculty of Computer Science

## CSCI 5308 - Quality Assurance

# Project Report

Scrawl : A Note-Taking Android Application

**Submitted By**

Group 19

**Members**

**B00#**

Aman Tewary

B00782684

Haofan Hou

B00783052

Nikhil Kamath

B00779777

**Submitted To**

Robert Hawkey

Kirstie Hawkey

**Date**

3<sup>rd</sup> August, 2018

# Table Of Content

<b>Objective</b>	<b>3</b>
<b>Technology Used</b>	<b>3</b>
<b>Workflow</b>	<b>4</b>
<b>Code Review</b>	<b>5</b>
<b>Continuous Integration &amp; Delivery</b>	<b>6</b>
<b>Logging and Crashlytics</b>	<b>10</b>
<b>Product Backlog</b>	<b>16</b>
<b>Sprint Details</b>	<b>17</b>
<b>Team Roles:</b>	<b>18</b>
Sprint A (2 Weeks)	18
Sprint B (2 Weeks)	18
Sprint C (2 Weeks)	19
Sprint D (2 Weeks)	19
<b>Code Contribution:</b>	<b>19</b>
<b>Separation of Logic</b>	<b>24</b>
<b>Examples of Refactoring</b>	<b>24</b>
<b>Design Patterns</b>	<b>28</b>
<b>Tests:</b>	<b>30</b>
<b>Functionalities:</b>	<b>32</b>
<b>Technical Debts</b>	<b>32</b>
<b>Custom Navigation Drawer</b>	<b>32</b>
<b>Configurable Business Logic</b>	<b>33</b>
<b>References</b>	<b>33</b>

## Objective

Scrawl is an Android application which allows the users to store their information within the app. The application requires Android OS version Marshmallow or above. This simple note taking application lets its users save their notes into their personal account. Moreover, they can archive their notes into different categories. For Instance, if the users want to take a note but want to add labels to archive it separately, they'll be able to do so in this application.

This application allows users to create an account by providing their email id and password. The credential received will be handled in a secure manner. After the account is created successfully, users are taken to the main screen where all their notes are kept. If the users don't have any note then the list will be empty. All the notes users take will be stored in a remote database. The users get to see a list of all the notes they have taken and have the ability to edit them anytime. Furthermore, Scrawl allows the users to select any text from anywhere and add it as a new note.

The application has different bells and whistles like it provide the ability to user to share their notes, in collaboration mode, with other users. The collaborated note won't have a delete option for the user who has receives the note from other users. Additionally, the application has an inbuilt bad-language filter which replaces bad words with an asterisk.

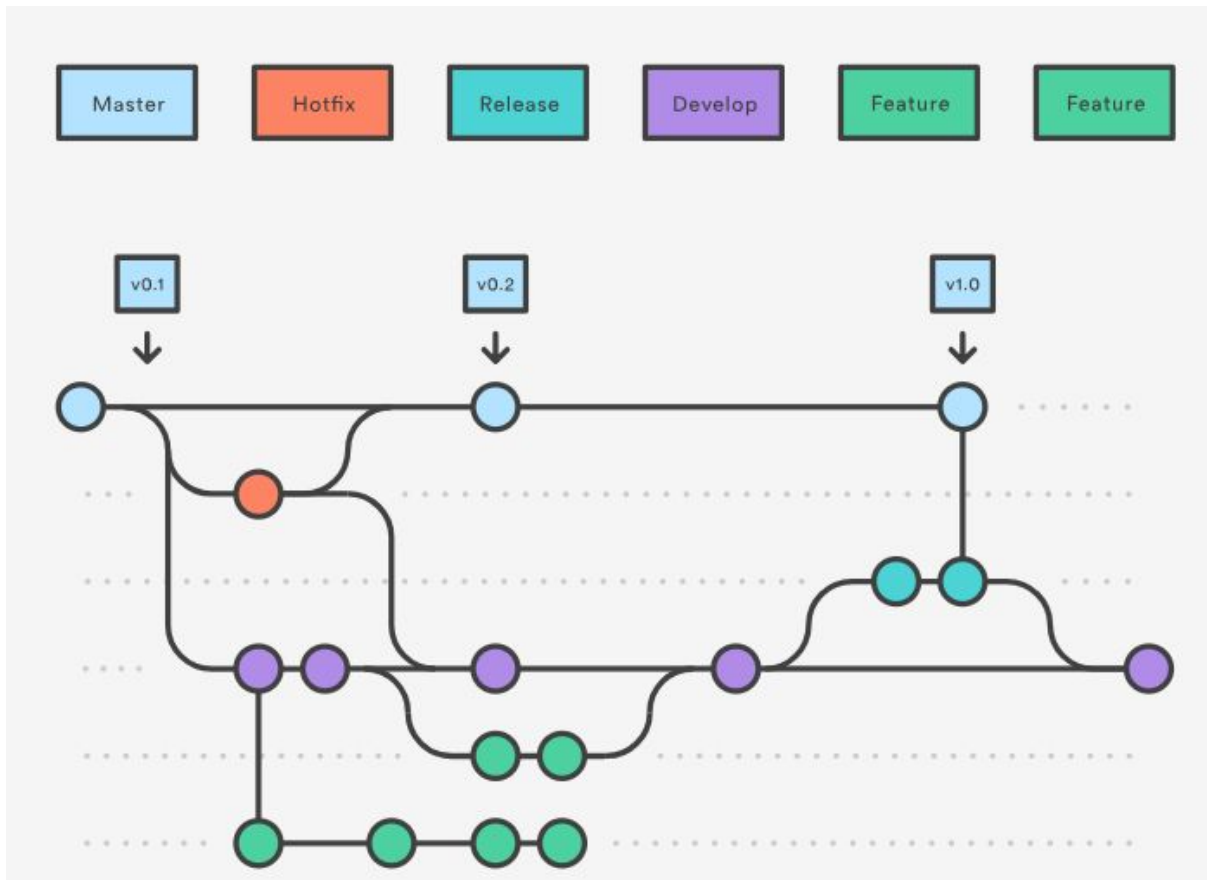
## Technology Used

- **Programming Languages**
  - Java (Android)
  - XML (Android)
  - PHP
  - JavaScript (Postman Test)
- **Tools**
  - PHPStorm
  - Android Studio
  - MySQL Workbench
  - Postman
- **Database**
  - MariaDB
- **Logging**
  - Firebase Crashlytics

- **Continuous Integration**
  - Jenkins
- **Testing**
  - JUnit4
  - Roboelectric
  - PHPUnit
  - AndroidJUnit4
  - Postman API Test Script (JavaScript)
  - Amazon Web Service Device Farm
- **Deployment**
  - Diawi (App Deployment)
  - Bluenose (API)
- **Team Management**
  - Trello
  - Slack

## **Workflow**

We followed GitFlow [1] workflow for our project by assigning very specific roles to different branches of our GitHub repository and defining how and when they should interact. However, instead of making multiple feature branches we decided to create individual develop branches for every member of our team. We created three core branches, namely master, test, and devint. The devint branch was used to push new features and run on our development environment. The test branch was to test our application with Unit Test, Instrumentation Test, and Monkey Test (AWS Device Farm)[2]. The master branch was our release branch which was used to deploy our finished product to diawi.com [3]. We do have some hot fix branches.



**Fig. 1 GitFlow HotFix Branches [1]**

## Code Review

For our project, we made our core branches (master, test, and devint) protected on GitHub so that each an every pull request will be reviewed by at least one other member of the team before merging it.

## Pull Request Review Sample

Hotfix Registration bug #28

Merged amantewary merged 1 commit into devint from dev-nikhil 17 days ago

Conversation 1 Commits 1 Checks 0 Files changed 1 +16 -0

Svenvollfied25 commented 17 days ago  
No description provided.

Hotfix - registration bug 3b8a694

Svenvollfied25 self-assigned this 17 days ago

Svenvollfied25 requested review from amantewary and HowfunHoo 17 days ago

HowfunHoo commented 17 days ago  
Looks good

amantewary approved these changes 17 days ago

amantewary merged commit 77fa93e into devint 17 days ago

Reviewers: amantewary (checked), HowfunHoo (pending)

Assignees: Svenvollfied25

Labels: None yet

Projects: None yet

Milestone: No milestone

Notifications: Unsubscribe

## Continuous Integration & Delivery

As we were working as a team, we needed to set up an environment where any member of the team can push their code to our core branches and it automatically builds the application, tests it and finally deploys it for public use.

We decided to setup Jenkins [4] which helped us to automate the non-human part of our application development process, with integration and facilitating technical aspects of continuous delivery.

We setup three jobs in our Jenkins server with specific roles in association with our GitHub core branches.



The screenshot shows a Jenkins job list table with columns: S, W, Name, Last Success, Last Failure, and Last Duration. There are three jobs listed: Group19, Group19-RELEASE, and Group19-TEST. Each job has a status icon, a weather icon, and a RSS icon. The table also includes a legend for the RSS icons.

S	W	Name	Last Success	Last Failure	Last Duration
		Group19	23 hr - #118	1 day 14 hr - #117	2 min 21 sec
		Group19-RELEASE	8 days 17 hr - #5	N/A	1 min 25 sec
		Group19-TEST	8 days 17 hr - #9	8 days 18 hr - #7	1 min 25 sec

Icon: S M L

Legend RSS for all RSS for failure

### Job #1: Group 19

This job was responsible for building the application after any member of our team merge pull request to the devint branch. This job checks for any new merge every 2 minutes. If the build for any reason it sends out the mail to each member of the group along with the details on what was the reason for the failure. In this job we are also generating lint reports.

### Successful Build Example:

```
Wrote XML report to file:///C:/Program%20Files%20(x86)/Jenkins/workspace/Group19/app/build/reports/lint-results.xml
:app:preDebugUnitTestBuild UP-TO-DATE
:app:javaPreCompileDebugUnitTest UP-TO-DATE
:app:compileDebugUnitTestJavaWithJavac
:app:generateDebugUnitTestConfig UP-TO-DATE
:app:processDebugUnitTestJavaRes NO-SOURCE
:app:testDebugUnitTest
:app:preReleaseUnitTestBuild UP-TO-DATE
:app:javaPreCompileReleaseUnitTest
:app:compileReleaseUnitTestJavaWithJavac
:app:generateReleaseUnitTestConfig UP-TO-DATE
:app:processReleaseUnitTestJavaRes NO-SOURCE
:app:testReleaseUnitTest
:app:test
:app:check
:app:build

BUILD SUCCESSFUL in 1m 55s
63 actionable tasks: 16 executed, 47 up-to-date
Build step 'Invoke Gradle script' changed build result to SUCCESS
```

## Failed Build Example:

```
at org.gradle.api.internal.tasks.compile.NormalizingJavaCompiler.delegateAndHandleErrors(NormalizingJavaCompiler.java:98)
at org.gradle.api.internal.tasks.compile.NormalizingJavaCompiler.execute(NormalizingJavaCompiler.java:51)
at org.gradle.api.internal.tasks.compile.NormalizingJavaCompiler.execute(NormalizingJavaCompiler.java:37)
at org.gradle.api.internal.tasks.compile.CleaningJavaCompilerSupport.execute(CleaningJavaCompilerSupport.java:35)
at org.gradle.api.internal.tasks.compile.incremental.SelectiveCompiler.execute(SelectiveCompiler.java:61)
at org.gradle.api.internal.tasks.compile.incremental.SelectiveCompiler.execute(SelectiveCompiler.java:34)
at org.gradle.api.internal.tasks.compile.incremental.IncrementalCompilationFinalizer.execute(IncrementalCompilationFinalizer.java:39)
at org.gradle.api.internal.tasks.compile.incremental.IncrementalCompilationFinalizer.execute(IncrementalCompilationFinalizer.java:24)
at org.gradle.api.tasks.compile.JavaCompile.performCompilation(JavaCompile.java:207)
at org.gradle.api.tasks.compile.JavaCompile.compile(JavaCompile.java:133)
at com.android.build.gradle.tasks.factory.AndroidJavaCompile.compile(AndroidJavaCompile.java:125)
at org.gradle.internal.reflect.JavaMethod.invoke(JavaMethod.java:73)
at org.gradle.api.internal.project.taskfactory.IncrementalTaskAction.doExecute(IncrementalTaskAction.java:46)
at org.gradle.api.internal.project.taskfactory.StandardTaskAction.execute(StandardTaskAction.java:39)
at org.gradle.api.internal.project.taskfactory.StandardTaskAction.execute(StandardTaskAction.java:26)
at org.gradle.api.internal.tasks.execution.ExecuteActionsTaskExecuter$1.run(ExecuteActionsTaskExecuter.java:121)
at org.gradle.internal.progress.DefaultBuildOperationExecutor$RunnableBuildOperationWorker.execute(DefaultBuildOperationExecutor.java:336)
at org.gradle.internal.progress.DefaultBuildOperationExecutor$RunnableBuildOperationWorker.execute(DefaultBuildOperationExecutor.java:328)
at org.gradle.internal.progress.DefaultBuildOperationExecutor.execute(DefaultBuildOperationExecutor.java:199)
at org.gradle.internal.progress.DefaultBuildOperationExecutor.run(DefaultBuildOperationExecutor.java:110)
at org.gradle.api.internal.tasks.execution.ExecuteActionsTaskExecuter.executeAction(ExecuteActionsTaskExecuter.java:110)
at org.gradle.api.internal.tasks.execution.ExecuteActionsTaskExecuter.executeActions(ExecuteActionsTaskExecuter.java:92)
... 29 more

* Get more help at https://help.gradle.org

BUILD FAILED in 28s
57 actionable tasks: 21 executed, 36 up-to-date
Build step 'Invoke Gradle script' changed build result to FAILURE
Build step 'Invoke Gradle script' marked build as failure
Not sending mail to unregistered user nikhilvk91@gmail.com
Sending e-mails to: aman.tewary@live.com nk634683@dal.ca haofan.hou@gmail.com
Finished: FAILURE
```

## Lint Report Sample:

Lint Report: 78 warnings

Check performed at Wed Jul 25 17:24:56 ADT 2018

Overview

Correctness

1

▲

CommitPrefEdits

Missing commit() on SharedPreferences.Editor

2

▲

DefaultLocale

Implied default locale in case conversion

1

▲

OldTargetApi

Target SDK attribute is not targeting latest version

3

▲

UnusedAttribute

Attribute unused on older versions

2

▲

GradleDependency

Obsolete Gradle Dependency

Security

1

▲

ExportedReceiver

Receiver does not require permission

1

▲

AllowBackup

AllowBackup/FullBackup/Content Problems

Performance

2

▲

ObsoleteSdkInt

Obsolete SDK\_INT Version Check

1

▲

UseCompoundDrawables

Node can be replaced by a TextView with compound drawables

1

▲

ViewHolder

ViewHolder Candidates

1

▲

InefficientWeight

Inefficient layout weight

1

▲

Overdraw

Overdraw: Painting regions more than once

27

▲

UnusedResources

Unused resources

Usability Icons

1

▲

IconColors

Icon colors do not follow the recommended visual style

3

▲

IconDensities

Icon densities validation

Usability

1

▲

GoogleAppIndexingWarning

Missing support for Firebase App Indexing

1

▲

TextFields

Missing InputType

Accessibility

3

▲

ContentDescription

Image without contentDescription

1

▲

LabelFor

Missing accessibility label

## Job #2: Group 19-TEST

This job was responsible for building the application after any member of our team merge pull request to the test branch. Its specific role is to sign the .apk (Android Application Package) file generated after the build is successfully finished. After the apk file is signed successfully it is sent to Amazon Web Service (AWS) Device Farm to carry out monkey testing on multiple devices with different versions of Android OS. AWS Device Farm is an app testing service that lets you test Android, iOS, and web apps on many devices at once, or

reproduce issues on a device in real time. Our device pool is mentioned in the following screenshot. AWS Device Farm also provides a short video of the test carried out along with the timed log with an event or error details.

## Device Pool

Devices

Device	OS	Test results	Total minutes
✓ Galaxy S8 Unlocked	7.0	✓ 3	00:03:08
✓ Google Pixel	7.1.2	✓ 3	00:03:01
✓ Google Pixel 2	8.0.0	✓ 3	00:03:18
✓ Google Pixel 2 XL	8.0.0	✓ 3	00:03:22
✓ Motorola Nexus 6	7.0	✓ 3	00:04:03
✓ Samsung Galaxy S9+ (Unlocked)	8.0.0	✓ 3	00:03:37

## AWS Device Farm Setup

Device Farm Project: Group19 Runs & sessions Learn more about unlimited testing Aman Support

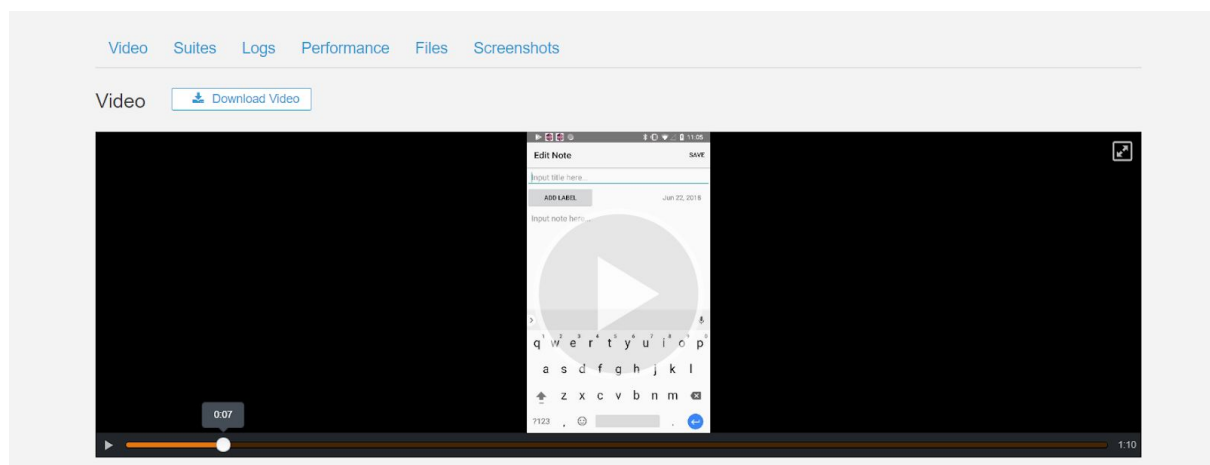
Automated tests Remote access Project settings

Automated runs allow you to execute built-in tests or your own scripts on one or more devices in parallel, generating a comprehensive report that includes high-level results, logs, screenshots, and performance data.

Create a new run

Run	Test results	Test Type	Created	Total minutes
! jenkins-app-rel...	1.1 1 16	Built-In: Explorer	2018-07-25T19:10-0300	00:26:03
✓ app-release (1...	18	Built-In: Explorer	2018-07-25T18:32-0300	00:20:33
jenkins-Group...	4 14	Built-In: Explorer	2018-06-23T20:14-0300	00:37:54

## AWS Device Farm Test Video Sample





## AWS Device Farm Log

Logs					
<a href="#">Download logs</a>					
Source	Time	PID	Level	Tag	Message
Device	00:46.591	1128	Info	ActivityManager	Start proc 16333:com.android.documentsui/u0a63 for broadcast com.androi...
Device	00:46.657	1128	Info	ActivityManager	Start proc 16350:com.google.android.apps.photos/u0a118 for broadcast co...
Device	00:46.695	16333	Info	ProvidersCache	Provider returned no roots. Possibly naughty: com.google.android.apps.doc...
Device	00:46.744	16350	Info	zygote64	Do partial code cache collection, code=0B, data=15KB
Device	00:46.744	16350	Info	zygote64	After code cache collection, code=0B, data=15KB
Device	00:46.744	16350	Info	zygote64	Increasing code cache capacity to 128KB
Device	00:46.787	1128	Info	ActivityManager	Killing 4225:com.android.keychain/1000 (adj 906): empty for 25619s
Device	00:46.788	1128	Warning	zygote64	kill(-4225, 9) failed: No such process
Device	00:46.823	5111	Info	LocationSettingsChecker	Removing dialog suppression flag for package com.android.chrome
Device	00:46.829	5111	Info	Icing	doRemovePackageData com.android.chrome
Device	00:46.829	5111	Info	Icing	Removing corpus key 4C4179C9B8DC2C610BC2FCFAFE2FF64D7D74A79...
Device	00:46.833	1128	Warning	zygote64	kill(-4225, 9) failed: No such process
Device	00:46.833	1128	Info	zygote64	Successfully killed process cgroup uid 1000 pid 4225 in 45ms
Device	00:46.904	5111	Info	Icing	updateResources: need to parse wzf{com.android.chrome}
Device	00:46.972	5111	Info	Icing	Indexing 4C4179C9B8DC2C610BC2FCFAFE2FF64D7D74A790 from com.a...

### Job #3: Group 19-RELEASE

This job was responsible for building the application as we finalize to release and deploy the application. The specific job of this job is to deploy the signed apk to diwai.com [3]. Diawi is a tool for developers to deploy Development and In-house applications directly to the devices. However, the issue with diwai is that for a free account the link to application changes after every Jenkins build and the deployed application expires in 5 days. However, we have provided supporting screenshots below from Jenkins build and Diwai Portal.

### Jenkins Release Job Deployment :

```
BUILD SUCCESSFUL in 1m 15s
64 actionable tasks: 23 executed, 41 up-to-date
Build step 'Invoke Gradle script' changed build result to SUCCESS
[Group19-RELEASE] $ cmd.exe /C "echo '""resolving effective environment""' && exit %ERRORLEVEL%"
[SignApksBuilder] searching environment variable ANDROID_HOME=D:\Android\Sdk for zipalign...
[SignApksBuilder] found zipalign in Android SDK's latest build tools: D:\Android\Sdk\build-tools\28.0.1\zipalign.exe
[SignApksBuilder] D:\Android\Sdk\build-tools\28.0.1\zipalign.exe -f -p 4 "C:\Program Files (x86)\Jenkins\workspace\Group19-RELEASE\app\build\outputs\apk\release\app-release-unsigned.apk" "C:\Program Files (x86)\Jenkins\workspace\Group19-RELEASE\out\zipalign\aligned-app-release-unsigned-8894354667345236559.apk"
[Group19-RELEASE] $ D:\Android\Sdk\build-tools\28.0.1\zipalign.exe -f -p 4 "C:\Program Files (x86)\Jenkins\workspace\Group19-RELEASE\app\build\outputs\apk\release\app-release-unsigned.apk" "C:\Program Files (x86)\Jenkins\workspace\Group19-RELEASE\out\zipalign\aligned-app-release-unsigned-8894354667345236559.apk"
[SignApksBuilder] signing APK SignApksBuilder-out\zipalign\aligned-app-release-unsigned-8894354667345236559.apk
[SignApksBuilder] deleting previous signed APK C:\Program Files (x86)\Jenkins\workspace\Group19-RELEASE\app\build\outputs\apk\release\app-release.apk
[SignApksBuilder] signed APK app\build\outputs\apk\release\app-release.apk
[SignApksBuilder] archiving signed APK app\build\outputs\apk\release\app-release.apk
[SignApksBuilder] finished signing APKs
C:/Program Files (x86)/Jenkins/workspace/Group19-RELEASE/app/build/outputs/apk/release/app-release.apk is being uploaded ...
upload job is 37HjbPcalGwRdTL5tHCpTM8A7kceDflma9zUbUpPTR
used proxy host is
used proxy port is 0
used proxy protocol is
status 2000
message Ok
C:/Program Files (x86)/Jenkins/workspace/Group19-RELEASE/app/build/outputs/apk/release/app-release.apk have been uploaded successfully to diawi ...
hash DhWxhx
link https://install.diawi.com/DhWxhx
Finished: SUCCESS
```

## Diwai Deployment Portal:

The screenshot shows the Diwai Deployment Portal interface. At the top, the 'diawi' logo is on the left, and the user 'Aman Tewary' with email 'am858620@dal.ca' is on the right. Below the header, the 'Uploaded apps' section is visible. A filter bar on the left shows 'Scrawl' with package ID 'com.example.amantewary.scrawl'. The main area displays a list of four uploaded apps, all named 'Scrawl' with package ID 'com.example.amantewary.scrawl'. Each app entry shows its platform (Android), version (1.0), build (1), and upload time (1 day ago or 2 days ago). To the right of each app entry is a status box showing the app's status (Available), the number of installations left (25), and the expiration time (in 1 day or in 22 hours). The app IDs are YsJq5V, GQEnSp, xYHTwN, and 3U6bxT. The total number of apps is 19.

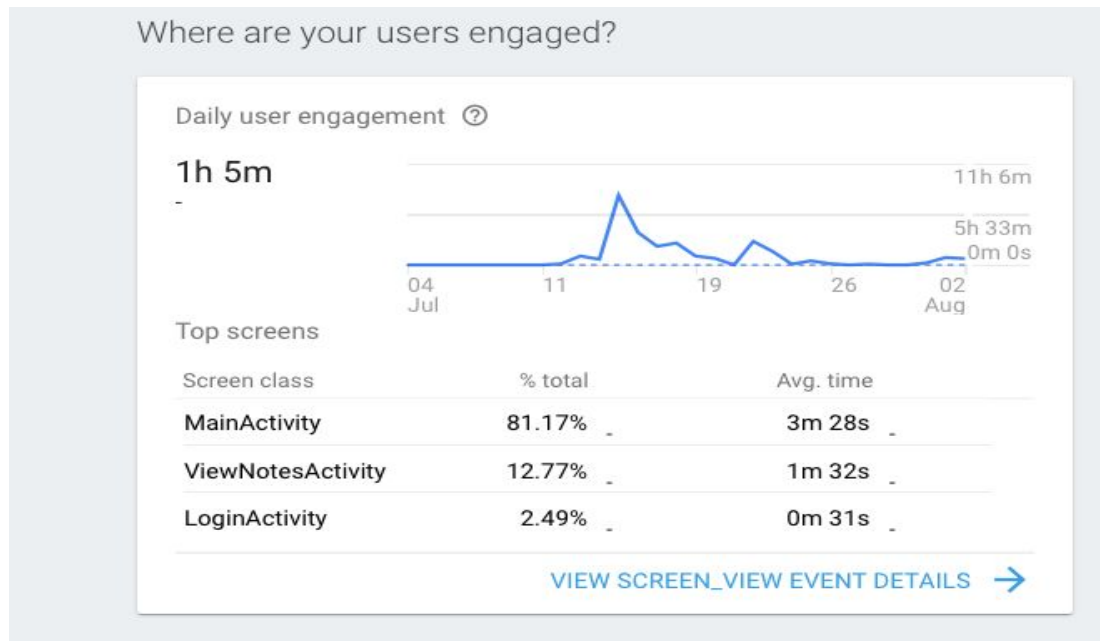
App Name	Package ID	Platform	Version	Build	Uploaded	Status	Installations Left	Expires
Scrawl	com.example.amantewary.scrawl	Android	1.0	1	1 day ago	Available	25	in 1 day
Scrawl	com.example.amantewary.scrawl	Android	1.0	1	1 day ago	Available	25	in 1 day
Scrawl	com.example.amantewary.scrawl	Android	1.0	1	1 day ago	Available	25	in 1 day
Scrawl	com.example.amantewary.scrawl	Android	1.0	1	2 days ago	Available	25	in 22 hours

## Logging and Crashlytics

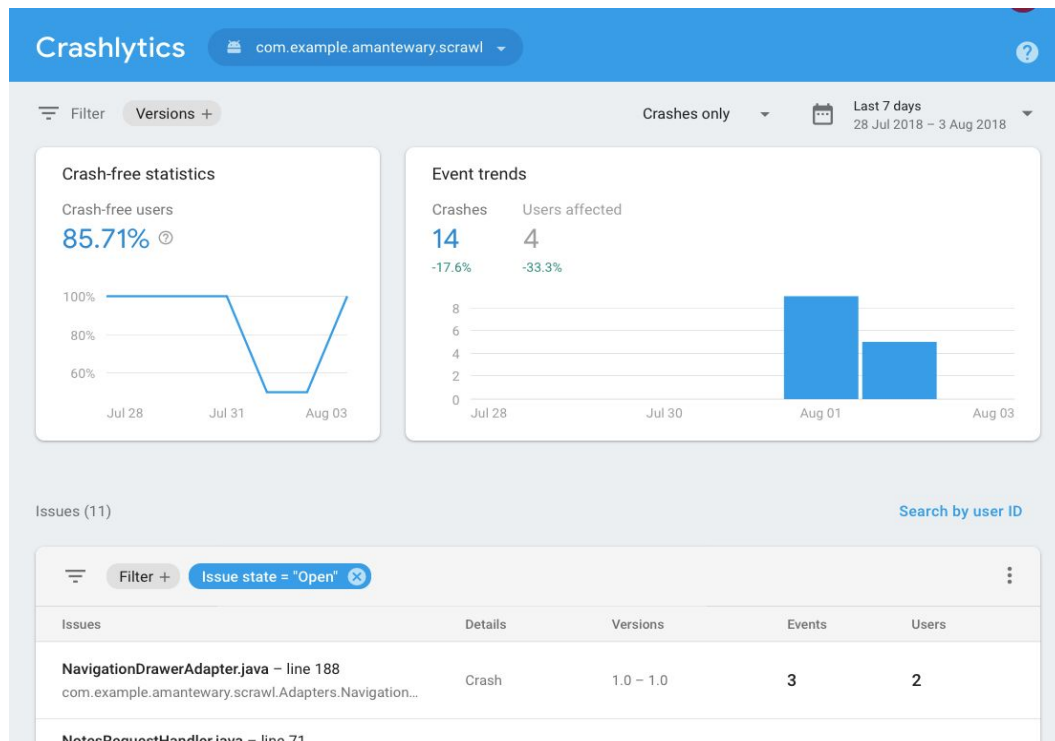
### 1. Firebase + Crashlytics



For logging and register user engagement with our application, we decided to use Firebase-SDK. Firebase<sup>[5]</sup> provides us with a dashboard where we can monitor the user's engagement with each activity. This helps us to understand where user spends most of their time on the app and maybe we can use that opportunity to add Advertisement or give priority to add new features in that activity.

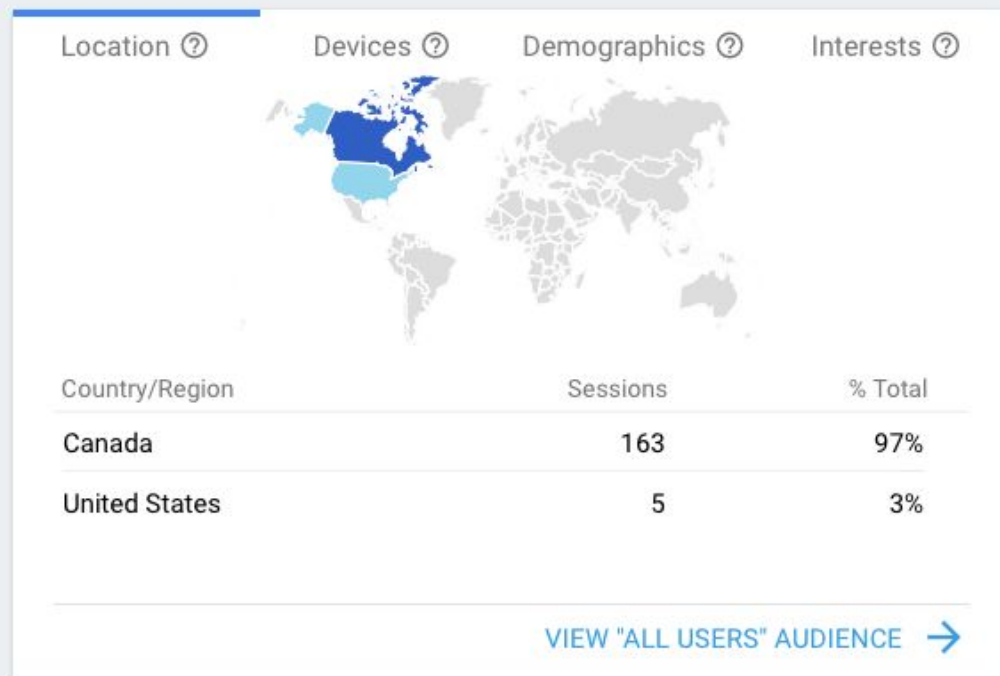


Similarly, we used crashlytics which catch the logs with the device name and time when the application crashes. This comes in handy because we don't have direct control of the application once it gets deployed the developers cannot make changes to the app. Also, since Android is a versatile platform with many different devices, there are more chances of the app getting crashed. These bugs often go unnoticed due to lack of testing devices in the company.



Additionally, we get other features like the location of users who are using our application. This feature lets us control the Locale and language preference if our application needs one. Combining this with AdMob<sup>[6]</sup> we can also choose to filter the type of advertisement we can show in our application. A really good example of this could be to ban the liquor advertisements in Muslim dominating countries since liquor is considered taboo in those countries.

What is your audience like?



## 2. Php (Backend) Logging

We also have implemented logger for our backend logic. We are using PHP to handle request from the device and generate a response in the form JSON which is read by the Android application using Retrofit library.

### Login And Registration API Log



```

Connection Opened at 02-08-2018 (Thu) 17:12:51 by test@test.com Request From 134.190.242.74
Connection Opened at 02-08-2018 (Thu) 17:12:51
Connection Closed at 02-08-2018 (Thu) 17:12:51
Connection Opened at 02-08-2018 (Thu) 17:14:23
Connection Opened at 02-08-2018 (Thu) 17:14:23 by test@test.com Request From 134.190.242.74
Connection Opened at 02-08-2018 (Thu) 17:14:23
Connection Closed at 02-08-2018 (Thu) 17:14:23
Connection Opened at 02-08-2018 (Thu) 17:31:00
Connection Opened at 02-08-2018 (Thu) 17:31:00 by test6@test.com Request From 134.190.229.233
Connection Opened at 02-08-2018 (Thu) 17:31:00
Connection Closed at 02-08-2018 (Thu) 17:31:00
Connection Opened at 02-08-2018 (Thu) 17:31:00
Connection Opened at 02-08-2018 (Thu) 17:31:00 by Request From 134.190.229.233
Connection Closed at 02-08-2018 (Thu) 17:31:00
Connection Opened at 02-08-2018 (Thu) 17:37:07
Connection Opened at 02-08-2018 (Thu) 17:37:07 by test@test.com Request From 134.190.242.74
Connection Opened at 02-08-2018 (Thu) 17:37:07
Connection Closed at 02-08-2018 (Thu) 17:37:07

```

File link: [https://web.cs.dal.ca/~kamath/QA\\_Devint/log.txt](https://web.cs.dal.ca/~kamath/QA_Devint/log.txt)

## HTTP Request Logger For Note API

```

2018-08-01 22:44:37 134.190.164.203 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-01 22:44:37 134.190.164.203 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-01 22:45:09 134.190.164.203 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-01 22:45:09 134.190.164.203 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-01 22:52:15 134.190.164.203 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-01 22:52:15 134.190.164.203 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-01 22:52:27 134.190.164.203 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-01 22:52:27 134.190.164.203 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-01 22:53:28 134.190.164.203 DELETE /~kamath/QA_Devint/NoteApi/v1/label/delete okhttp/3.10.0
2018-08-01 22:53:28 134.190.164.203 DELETE /~kamath/QA_Devint/NoteApi/v1/label/delete okhttp/3.10.0
2018-08-01 22:53:50 134.190.164.203 POST /~kamath/QA_Devint/NoteApi/v1/label/create okhttp/3.10.0
2018-08-01 22:53:50 134.190.164.203 POST /~kamath/QA_Devint/NoteApi/v1/label/create okhttp/3.10.0
2018-08-01 22:54:00 134.190.164.203 PUT /~kamath/QA_Devint/NoteApi/v1/label/update okhttp/3.10.0
2018-08-01 22:54:00 134.190.164.203 PUT /~kamath/QA_Devint/NoteApi/v1/label/update okhttp/3.10.0
2018-08-01 22:54:04 134.190.164.203 POST /~kamath/QA_Devint/NoteApi/v1/label/create okhttp/3.10.0
2018-08-01 22:54:04 134.190.164.203 POST /~kamath/QA_Devint/NoteApi/v1/label/create okhttp/3.10.0
2018-08-01 22:54:14 134.190.164.203 PUT /~kamath/QA_Devint/NoteApi/v1/label/update okhttp/3.10.0
2018-08-01 22:54:14 134.190.164.203 PUT /~kamath/QA_Devint/NoteApi/v1/label/update okhttp/3.10.0
2018-08-01 22:54:24 134.190.164.203 DELETE /~kamath/QA_Devint/NoteApi/v1/label/delete okhttp/3.10.0
2018-08-01 22:54:24 134.190.164.203 DELETE /~kamath/QA_Devint/NoteApi/v1/label/delete okhttp/3.10.0
2018-08-02 13:05:05 76.11.28.68 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-02 13:05:05 76.11.28.68 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-02 13:30:51 76.11.22.245 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-02 13:30:51 76.11.22.245 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-02 13:31:02 76.11.22.245 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-02 13:31:02 76.11.22.245 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-02 13:31:21 76.11.22.245 DELETE /~kamath/QA_Devint/NoteApi/v1/label/delete okhttp/3.10.0
2018-08-02 13:31:21 76.11.22.245 DELETE /~kamath/QA_Devint/NoteApi/v1/label/delete okhttp/3.10.0
2018-08-02 13:31:29 76.11.22.245 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-02 13:31:29 76.11.22.245 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-02 13:31:40 76.11.22.245 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-02 13:31:40 76.11.22.245 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-02 13:32:04 76.11.22.245 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0
2018-08-02 13:32:04 76.11.22.245 GET /~kamath/QA_Devint/NoteApi/v1/label/read?user_id=12 okhttp/3.10.0

```

File Link: [https://web.cs.dal.ca/~kamath/QA\\_Devint/NoteApi/includes/request.log](https://web.cs.dal.ca/~kamath/QA_Devint/NoteApi/includes/request.log)

## Tracker for Login and Registration

```
Time: 02-08-2018 (Thu) 16:57:33      Request Agent okhttp/3.10.0
Request Method POST
Requested at 1533229053
Connection Status 0

Time: 02-08-2018 (Thu) 17:12:51      Request Agent okhttp/3.10.0
Request Method POST
Requested at 1533229971
Connection Status 0

Time: 02-08-2018 (Thu) 17:14:23      Request Agent okhttp/3.10.0
Request Method POST
Requested at 1533230063
Connection Status 0

Time: 02-08-2018 (Thu) 17:31:00      Request Agent PostmanRuntime/7.1.1
Request Method POST
Requested at 1533231060
Connection Status 0

Time: 02-08-2018 (Thu) 17:31:00      Request Agent PostmanRuntime/7.1.1
Request Method POST
Requested at 1533231060
Connection Status 0

Time: 02-08-2018 (Thu) 17:37:07      Request Agent okhttp/3.10.0
Request Method POST
Requested at 1533231427
Connection Status 0
```

File Link: [https://web.cs.dal.ca/~kamath/OA\\_Devint/tracker.txt](https://web.cs.dal.ca/~kamath/OA_Devint/tracker.txt)


## Error/Event Logging for Note API

```
[02-Aug-2018 17:37:12 UTC] -----<Database Connection Closed>-----
[02-Aug-2018 17:37:25 UTC] -----<Database Connection Open>-----
[02-Aug-2018 17:37:25 UTC] Request Access To Create Label By User ID: 12
[02-Aug-2018 17:37:25 UTC] Invoked create() Method Inside Labels Class
[02-Aug-2018 17:37:25 UTC] Label Created With Name: LabelBy User: 12
[02-Aug-2018 17:37:25 UTC] -----<Database Connection Closed>-----
[02-Aug-2018 17:37:32 UTC] -----<Database Connection Open>-----
[02-Aug-2018 17:37:32 UTC] Request Access To Update Labels By User ID: 12with old label name Label & new label name Lab
[02-Aug-2018 17:37:32 UTC] Label Updated
[02-Aug-2018 17:37:32 UTC] -----<Database Connection Closed>-----
[02-Aug-2018 17:37:37 UTC] -----<Database Connection Open>-----
[02-Aug-2018 17:37:37 UTC] Request to Access Note ID: 128
[02-Aug-2018 17:37:37 UTC] Invoked read_single() Method
[02-Aug-2018 17:37:37 UTC] Retrieved Note
[02-Aug-2018 17:37:37 UTC] -----<Database Connection Closed>-----
[02-Aug-2018 17:37:40 UTC] -----<Database Connection Open>-----
[02-Aug-2018 17:37:40 UTC] Request to Access Note ID: 128
[02-Aug-2018 17:37:40 UTC] Invoked read_single() Method
[02-Aug-2018 17:37:40 UTC] Retrieved Note
[02-Aug-2018 17:37:40 UTC] -----<Database Connection Closed>-----
[02-Aug-2018 17:37:43 UTC] -----<Database Connection Open>-----
[02-Aug-2018 17:37:43 UTC] Request Access To Update Notes By User ID: 12
[02-Aug-2018 17:37:43 UTC] Note Updated
[02-Aug-2018 17:37:43 UTC] -----<Database Connection Closed>-----
[02-Aug-2018 17:37:44 UTC] -----<Database Connection Open>-----
[02-Aug-2018 17:37:44 UTC] Request to Access Note ID: 128
[02-Aug-2018 17:37:44 UTC] Invoked read_single() Method
[02-Aug-2018 17:37:44 UTC] Retrieved Note
[02-Aug-2018 17:37:44 UTC] -----<Database Connection Closed>-----
[02-Aug-2018 17:38:05 UTC] -----<Database Connection Open>-----
[02-Aug-2018 17:38:05 UTC] Request to Access To Delete Notes ID: 128
[02-Aug-2018 17:38:05 UTC] Note Deleted with ID: 128
[02-Aug-2018 17:38:05 UTC] -----<Database Connection Closed>-----
```

File Link: [https://web.cs.dal.ca/~kamath/QA\\_Devint/NoteApi/includes/debug.log](https://web.cs.dal.ca/~kamath/QA_Devint/NoteApi/includes/debug.log)

## Product Backlog

After submitting our project proposal we created a product backlog and took an internal vote for each an every user story and assigned story points. Based on the story points we started working on the stories.

 Product Backlog

File Edit View Insert Format Data Tools Add-ons Help

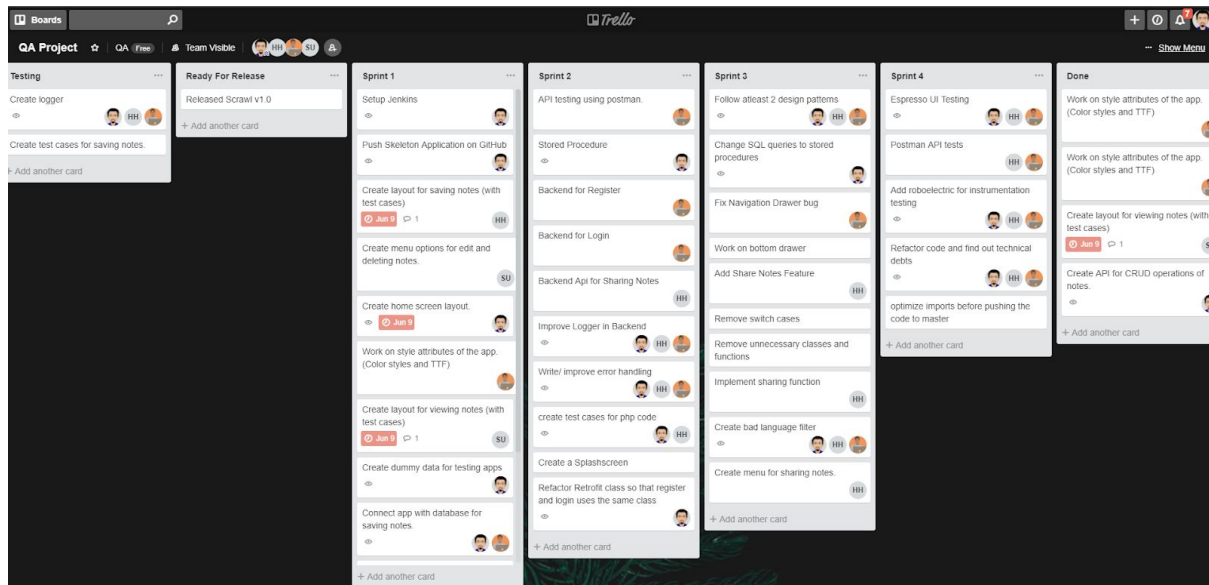
fx |

	A	B	C	D	E	F	G
	Importance	Story	Story Points	Team Member 1 vote	Team Member 2 vote	Team Member 3 vote	Team Member 4 vote
2	1	As a user, I want to see list of all my notes so that I can access my notes as soon as I open the application.	1.75	1	1	3	2
3	2	As a user, I want to create a new note and save it so that I can retrieve that note at a later point	4.5	3	5	5	5
4	3	As a user, I want to save links of blog, post and videos that I found online so that I can access that content at a later point.	3	2	1	5	4
5	4	As a user, I want to view a specific note so that I can read its content.	2	2	1	3	2
6	5	As a user, i want to register for an account so that I can sign up for the account	3.75	4	4	3	4
7	6	As a user, I want a login page so that I can securely sign in into my account and retrieve all my notes.	3.75	4	4	3	4
8	7	As a user, I want to edit a note so that I can update or fix errors in an existing note	3.5	3	4	4	3
9	8	As a user, I want to delete a note so that I can discard not I no longer want.	2.25	3	2	1	3
10	9	As a user, I want to search all my notes so that I can retrieve a specific note.	3.5	2	2	5	5
11	10	As a user, I want to label my notes so that I can keep it organized and easily accessible.	3.25	3	1	5	4
12	11	As a user, I want to add new labels so that I can categorize notes any way I want.	3.25	3	4	3	3
13	12	As a user, I want to share my notes with others so that they can view its content	4.5	4	4	5	5



## Sprint Details

We used Trello for managing our sprints. We divided our work in four sprints.



### Sprint 1:

- Setting up the development environment for the Android application.
- Setting up the development environment for PHP backend application.
- Setting up continuous integration using Jenkins.
- Working on XML layouts for Android Application.
- Working on API (PHP) for CRUD operation of notes.

### Sprint 2:

- Working on API (PHP) for CRUD operation of labels.
- Working on the Android application (JAVA) for CRUD operation of notes.
- Working on API (PHP) for user login and registration.
- Working on the Android application (JAVA) for User Login and Registration.
- Adding Error Handling and loggers for all the backend API.

### Sprint 3:

- Adding firebase crashlytics for error and event logging for Android Application.
- Setting up deployment environment on Jenkins.
- Setting up deployment on Diawi portal.
- Refactoring code for improving adding design patterns.
- Working on PHPUnit test cases for all the API's.
- Working on Postman API test.
- Working on note sharing API (PHP).
- Working on Android application activities (JAVA) for implementing the sharing feature.

#### Sprint 4:

- Working on Android activities (JAVA) for Adding, editing and deleting labels.
- Working on espresso tests for android activities.
- Working on Postman API tests.
- Working on Unit Test for Android POJO classes.
- Working on Unit Test for PHP backend API.
- Optimizing imports for Java code.

#### Team Roles:

Sprint A (2 Weeks)
--------------------

Members	Roles
Aman Tewary	Scrum Master
Nikhil Kamath	Developer
Haofan Hou	Developer/Tester

Sprint B (2 Weeks)
--------------------

Members	Roles
Aman Tewary	Developer
Nikhil Kamath	Scrum Master
Haofan Hou	Developer

### Sprint C (2 Weeks)

Members	Roles
Aman Tewary	Developer/Scrum Master
Nikhil Kamath	Developer/Tester
Haofan Hou	Developer

### Sprint D (2 Weeks)

Members	Roles
Aman Tewary	Developer/Tester
Nikhil Kamath	Developer/Tester
Haofan Hou	Scrum Master

### Code Contribution:

Team Member	Feature	Android/PHP	Classes Activities & Methods
Aman Tewary	Notes CRUD (Backend & Frontend)	Android	ICreateNote
			IDeleteNote
			IGetNoteByLabel
			INoteResponse
			IUpdateNote
			ViewNotesBaseActivity
			AddNotesActivity

			EditNotesActivity
			FilteredNotesActivity
			MainActivity <ul style="list-style-type: none"> <li>· populateNotesList()</li> <li>· initialLabelListLoading()</li> </ul>
			NotesListAdapter
			ViewHolder
			FilteredNotesActivityAdapter
			NoteHandler
			NoteRequestHandler <ul style="list-style-type: none"> <li>· createNotes()</li> <li>· getSingleNote()</li> <li>· getNotesListByLabel()</li> <li>· editNotes()</li> <li>· deleteNotes()</li> </ul>
			ViewNotesActivity <ul style="list-style-type: none"> <li>· deleteNotes()</li> </ul>
		PHP	ConnectDb.php
			Notes.php
			create.php
			delete.php
			read_single.php
			read.php
			readbyLabel.php
			readByUser.php
			update.php

	Labels CRUD (Backend and Frontend)	Android	ICreateLabel
			IDeleteLabel
			IGetLabels
			ILabelResponse
			LabelHandlers
			SharedPreferencesHandler
			LabelLoader
			LabelRequesthandler
		PHP	Labels.php
			create.php
	delete.php		
	update.php		
	read.php		
	Notes Reminder	Android	ViewNotesActivity ·     onTimeSet()
			ViewShareddNotesActivity ·     onTimeSet()
			NotificationHandler
			NotificationReciever
			TimePicker
	Bad Word Filter (Change Bad Word to Asterisks)	Android	Inputhandler ·     Inputvalidator ·     InputCensor ·     InputErrorHandling
	Logger	PHP	HttpLogger.php
			Logger.php

Nikhil Kamath	Labels	Android	NavigationDrawerAdapter
			NavigationModel
			NavObserver
			MainActivity · loadLabelForList()
	Login & Registration	Android	ILoginUser
			IRegisterUser
			UserClass
			ActivityRegister
			AppURLs
			EmailPasswordValidation
			InputHandler · readTxt() · doRealTimeLanguageCheck()
			LoginActivity
			RetroFitInstance
			SessionManger
			MainActivity · showDialog() · signOut()
		PHP	Database_Queries.php
			Login.php
			Register.php
	Notes Search	Android	NotesListAdapter · getFilter()

	Crash Analytics	Android	MainActivity · Firebase
	Splash Screen	Android	SplashScreen
Haofan	Note Sharing	Android	ICreateShare
			IDeleteShare
			ICheckUser
			IGetNote
			ShareHandler
			SharedRequestHandler
			ViewNotesActivity · setSharedIntent() · showDialog() · setCollaborateInfo() · sendRequest()
			ViewSharedNoteActivity
			NoteContext
			NoteState
			SharedNote
			ViewNote
			MainActivity · swipeRefresh() · populateNotesList()
			UserRequestHandler
			IUserExistsResponse
			NoteRequestHandler · getAllNotesByUserId()

		PHP	Share.php
			Create.php
			Delete.php
			getAllNotesForUser.php

## Separation of Logic

Since we were developing in this project with scrutiny, we have made sure we try to separate the presentation layer, data layer and business layer. Android application development is beautiful because even though if someone is developing an application without even keeping design patterns in his/her head, the process of development will make sure that we follow some pattern.

In this project, we have used Interface Segregation and Single responsibility principle where we felt it was required. There's an entire folder full of interfaces in our project which keeps our project organized. Even our backend has separate files for getting GET or POST data and generate a response and calling stored procedures along with password hashing.

For instance, in Android when the user tries to add a new note, he opens the [AddNotesActivity.java](#) (Presentation layer) and writes something. The censor that checks every letter (Business Layer) is present in a separate file which is called [InputHandler.java](#). Finally when the user enters the post button, the file that sends the data to the backend is written in a separate file called [NotesRequestHandler.java](#).

## Examples of Refactoring

There were a lot of changes over the period of 3 weeks in both backend and Android application. We have many examples when we made changes and refactoring but to list a

### 1. Add Notes and Edit Notes (Extract Method)



```

public void inputErrorHandling(EditText title, EditText body, EditText link) {

    if (title.getText().toString().trim().matches("")) {
        title.setBackgroundResource(R.drawable.border_error);
        title.setError("Enter Note Title");
        return;
    } else {
        title.setBackgroundResource(R.drawable.border);
    }
    if (body.getText().toString().trim().matches("")) {
        body.setBackgroundResource(R.drawable.border_error);
        body.setError("Enter Note Body");
        return;
    } else {
        body.setBackgroundResource(R.drawable.border);
    }
    if (!Patterns.WEB_URL.matcher(link.getText().toString().trim()).matches()) {
        link.setError("Please Enter Valid URL");
        return;
    } else {
        link.setBackgroundResource(R.drawable.border);
    }
}
}

```

### InputHandler.java

The image above shows a function that was inside the functions shown in the image below.

```

public void addNote() {

    try {
        label = inputHandler.inputCensor(sp_add_labels.getSelectedItem().toString());
        title = inputHandler.inputCensor(et_title.getText().toString().trim());
        body = inputHandler.inputCensor(et_content.getText().toString().trim());
        link = et_link.getText().toString().trim();
        String status = "(created)";
        String date = tv_date.getText().toString();
        //TODO: Need to change user_id once login and registration is done.
        NoteHandler noteHandler = new NoteHandler(label, title, body, link, Integer.parseInt(sessionManager.getUserDetails));
        if (inputHandler.inputValidator(title, body, link)) {
            NotesRequestHandler request = new NotesRequestHandler();
            request.createNote(noteHandler, AddNotesActivity.this);
        } else {
            inputHandler.inputErrorHandling(et_title, et_content, et_link);
        }
    } catch (Exception e) {
        Log.e(TAG, "Message: " + e.toString());
    }
}
}

```

### AddNoteActivity.java

```

try {
    sessionManager = new SessionManager(getApplicationContext());
    NoteHandler noteHandler = new NoteHandler(noteId, label, title, body, link, Integer.parseInt(sessionManager.getUserSessionId()));
    if (inputHandler.inputValidator(title, body, link)) {
        request.editNote(noteHandler, EditNotesActivity.this);
    } else {
        inputHandler.inputErrorHandling(et_title, et_content, et_link);
    }
} catch (Exception e) {
    Log.e(TAG, "Message: " + e.toString());
}
}

```

EditNoteActivity.java

## 2. Sharedpreference (Long parameter list)

```

public void createLoginSession(String name, String email){
    // Storing login value as TRUE
    editor.putBoolean(IS_LOGIN, true);

    // Storing name in pref
    editor.putString(KEY_NAME, name);

    // Storing email in pref
    editor.putString(KEY_EMAIL, email);

    // commit changes
    editor.commit();
}

```

The image above shows how we were using a long parameter inside createLoginSession() method (which was going to get bigger). We decided to send the object of User class as parameter instead of a many parameters. The image below shows how we refactored it.

```

public void createLoginSession(UserClass userClass){

    // Storing login value as TRUE
    this.userClass = userClass;
    editor.putBoolean(IS_LOGIN, true);

    // Storing name in pref
    editor.putString(KEY_NAME, userClass.getUsername());

    // Storing email in pref
    editor.putString(KEY_EMAIL, userClass.getEmail());

    editor.putInt(KEY_USERID, userClass.getUserId());

    // commit changes
    editor.commit();
}

```

### 3. doRealtimeCheck (Method Extraction)

```

private void doRealTimeCheck() {
    et_content.addTextChangedListener(new TextWatcher() {
        @Override
        public void beforeTextChanged(CharSequence charSequence, int i, int i1, int i2) {

        }

        @Override
        public void onTextChanged(CharSequence charSequence, int i, int i1, int i2) {

        }

        @Override
        public void afterTextChanged(Editable editable) {
            if (editable.length() != 0) {
                inputHandler.inputCensor(et_content.getText().toString().trim());
            }
        }
    });

    et_title.addTextChangedListener(new TextWatcher() {
        @Override
        public void beforeTextChanged(CharSequence charSequence, int i, int i1, int i2) {

```

Previously this method was used check two edittexts and see whether they had bad word or not. This was happening in AddNotesActivity and EditNotesActivity. But then we decided to extract the method and wrap it around and the image below shows how we were able to achieve it.

```

public void doRealTimeLanguageCheck(final EditText editText) {
    try {
        editText.addTextChangedListener(new TextWatcher() {
            @Override
            public void beforeTextChanged(CharSequence charSequence, int i, int i1, int i2) {
                // TODO Auto-generated method stub
            }

            @Override
            public void onTextChanged(CharSequence charSequence, int i, int i1, int i2) {
                // TODO Auto-generated method stub
            }

            @Override
            public void afterTextChanged(Editable editable) {
                if (editable.length() != 0) {
                    inputCensor(editText.getText().toString().trim());
                }
            }
        });
    } catch (Exception e) {
        Log.e(TAG, "Message: " + e.toString());
    }
}
}

```

---

We decided to send the Edittext object instead which could be called in both AddNotesActivity and EditNotesActivity.

## Design Patterns

### 1. Singletons

We have implemented 4 Singletons patterns in this application. The use of each of them are explained below:

- **AppURL**

We are accessing the base url where we have php files to handle incoming information, via POST and GET, to generate JSON response. We decided to use singleton to store the base url so that it can be accessible within the app at the same time making sure that no one can modify it. This pattern can be easily found in the AppURLs.java file.

<https://github.com/DalhousieUniversityCSCI5308/Group19/blob/master/app/src/main/java/com/example/amantewary/scrawl/AppURLs.java>

- **Retrofit Instance**

Gone are the days when we could connect to a database through Android application via JDBC/ODBC connections. They were not secure and hence were deprecated. We have used Retrofit to parse the JSON response from our weburl. Retrofit converts the JSON response

into Java Objects. Since, the core retrofit was widely used throughout the application, we decided to make it a singleton to save us the hassle of rewriting it again and again.

This file shows the implementation of singleton [RetroFitInstance.java](#)

- **Label Loader**

The labels in our applications are stored in SharedPreferences. SharedPreferences are basically a database which is unstructured and non relational where people can store data in the OS which will work even when the app is closed or even uninstalled. They work like key value pairs and are not suitable for storing crucial information. We are using it to store the information about labels in this case. To access the SharedPreferences from anywhere, we have used Singleton. This class also lets us to add and fetch the list of labels and hence makes everything a lot easier.

The implementation can be found here in [LabelLoader.java](#)

## 2. Observer

- **The Bad-Word filter**

When the user tries to enter a content inside the EditText, a listener listens to the changes happening to the EditText and sends the text to a function which constantly checks whether the entered text is a bad abusive word or not. The Observer comes into the action when our bad language censor detects a bad word and notifies the activity which was letting the user to add notes. The activity reads the update and generates an Android toast saying “bad words will be censored”.

The files that showcase this pattern can be found in [InputHandler.java](#), [AddNotesActivity.java](#) and [EditNotesActivity.java](#)

- **Custom Navigation Drawer**

When the user opens the navigation drawer in the application, they can add new labels from the drawer by clicking the Add Label button. Since this is a custom navigation drawer and placed in a separate java file, there was no way to add a visual feedback that a new label is created. To notify the class that the button is clicked, we have used an observer that will notify the class which has the logic to handle the UI change.

The classes where we have implemented Observer pattern are [NavigationDrawerAdapter.java](#) and [NavObserver.java](#)

## 3. State

In this app, we have types of note - the notes created by the current user and the notes shared by others, and for each type of notes we hope to give user different level of permission to operate with the note. For example, for the owned notes, users are allowed to share the notes and delete the notes, while for the shared notes, they are not supposed to do so as the notes are not owned by themselves.

In this case, the app needs to run different activities with different UI and business logic for viewing notes. We regard the two types of notes as two states of notes. When users click a note on their note list, the app will direct them to the proper activity based on the state of the clicked note. The files used to implement State Pattern can be seen below:

- **NoteState**  
NoteState is an interface includes a method with which the app will direct user to the proper activity.
- **NoteContext**  
NoteContext is a class which stores and indicates the current state of the notes.
- **SharedNote**  
SharedNote is a class which implements NoteState interface and complete the codes for directing users to ViewSharedNoteActivity. ViewSharedNoteActivity is the activity for viewing shared notes.
- **ViewNote**  
ViewNote is a class which implements NoteState interface and complete the codes for directing users to ViewNotesActivity. ViewNotesActivity is the activity for viewing owned notes.

#### 4. Data Access Object (DAO)

Since this is an Android application, whenever we wanted to store any data from the response from the server, we have used DAO because that is how Retrofit needs to work. There are ample of examples in this project where we have DAOs. There's a separate folder where we have kept all the files which also helps in maintaining the project files.

This folder contains most of the DAOs. [Handlers](#).

## Tests

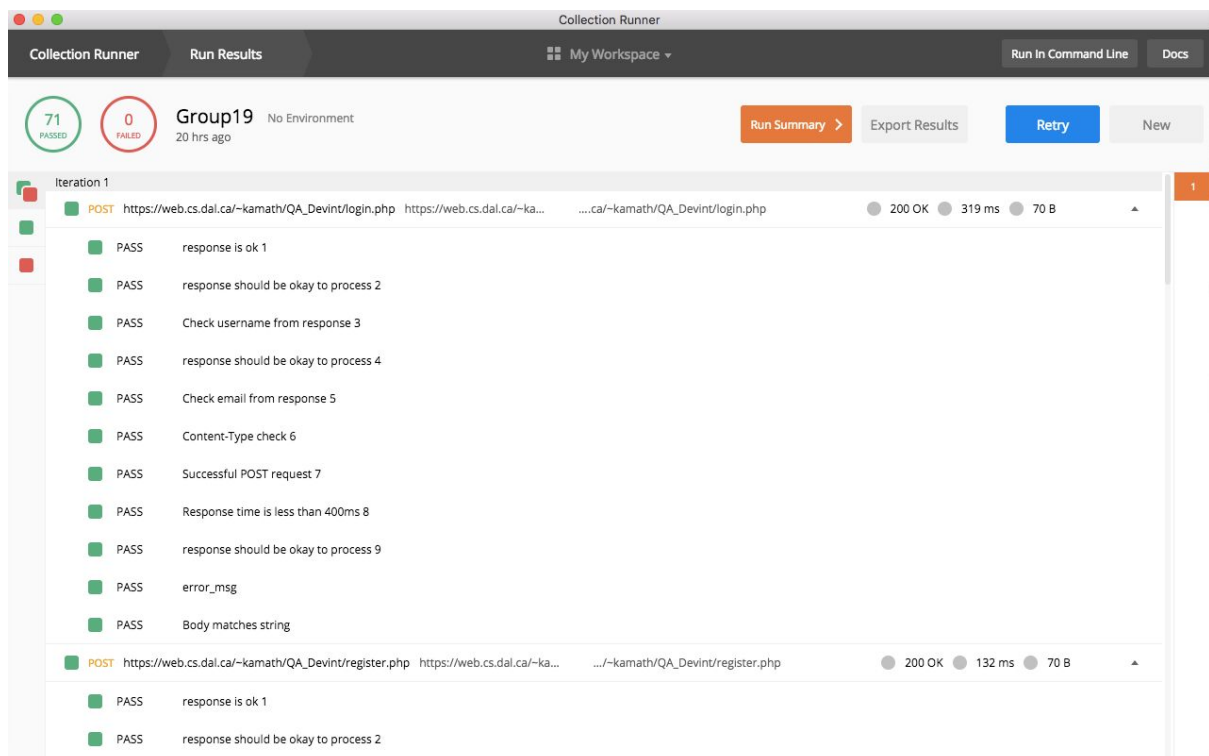
- **PHPUnit Test**

We created server-side API's for notes (CRUD), labels(CRUD), sharing notes, user login and registration using PHP. We wrote tests to cover every CRUD operation for notes and labels.

We are also covering PHP methods responsible for login and registration of the user. This helped us in writing error-free code for our data layer to avoid any unexpected crash.

- **Postman Test**

Apart from the PHP unit test cases, we also covered REST api tests using Postman. We tested all the endpoints where we generating the JSON response and wrote different tests cases to test the schema and whether we are actually receiving a the response we are expecting. For this particular test, we didn't follow TDD because it is impossible to test the responses without actually writing the code to generate the response.



- **AndroidJUnit4 Test (Espresso):**

We did espresso test for every function in our android application. With espresso test, we tried to cover UI as well as functionalities on a higher level. This step was taken mostly because most of our methods in Android have no return value and are interconnected together.

**NOTE:** Before running Espresso Tests (except for LoginTest and ActivityRegisterTest) you need to first login in the application. We know that it is not a good practice to add a condition for running test. However, to make it simple for evaluation we didn't mock the login. This was done to avoid a crash in case the user is already logged in during evaluation.

- **JUnit Test:**

We added the unit test for simple POJO classes in our android application. We didn't write many unit tests because most of the activities, classes, and methods are interconnected and it was impossible to break it into a singular unit. Therefore, we added an espresso test which combines multiple units together to test a single functionality.

## **Functionalities:**

- User Authentication
  - Login
  - Logout
  - Registration
- CRUD operations of notes
  - Creating notes
  - Adding link in a note
  - Updating notes
  - Deleting notes
  - Showing all notes in a list
  - Viewing the detail of a specific note
- Note searching and labeling
  - Searching for specific notes
  - Filtering notes by labels
  - Labeling notes
  - Creating labels and renaming labels
- Note Sharing
  - Sharing notes with third-party apps
  - Sharing notes within the app and collaborating on notes edit with other users
- Other functionalities
  - Setting reminder for notes
  - Bad language filter

## **Technical Debts**

- **Real-time collaboration**

We were planning to make the collaboration real-time, similar to google doc so that the users collaborating editing a note can see the changes made by others synchronously. However, since the project is required to use the MySQL database which is not a real-time database, it will be really hard to achieve this purpose. If we want the notes to be updated synchronously, the page needs to be refreshed every second at the back end, which will significantly reduce the performance of the app. Thus, we have to give up this plan. In the future, if we could move our database to Firebase real-time database, this function will be possible.



- **Custom Navigation Drawer**

This is the part where we had to make navigation drawer from scratch and then we had to make sure that users were able to edit the labels within the drawer. Since, we had little to none experience in working with navigation let alone, creating a custom one was quite a task. The code in the NavigationDrawerHandler is not at all modifiable and it's highly likely that the logic will break even if we had to new functionality in future. The entire code in that file is written to cater only logic and it's rigid and highly coupled.

- **Single Request Handler Class**

In our application we created four separate request handlers namely, NotesRequestHandler, LabelRequestHandler, ShareRequestHandler and UserRequestHandler. Even Though, each method inside these classes are specific role and in no way repetitive, there are certain part of code inside these methods that is repeating. We tried to rectify this using generic class and methods but was not successful. This is mainly because that part of code is inbuilt method of Retrofit. If had more time we could have tried to implement command pattern.

## **Configurable Business Logic**

Configurable business logic allows business users to make some changes to cater the needs of businesses without reprogramming or changing information in the application<sup>[7]</sup>. In our Android application, we have four labels which are fixed for all the users. Which means, all the users of our application can see those four labels which are “Personal”, “Important”, “Work” and “Reminder”. However, if we choose to add or rather remove a label from the application, we can do that directly from our side, without altering the application or releasing a new version.

Furthermore, we can also alter the language of these default labels depending the user's locale. New Labels with the user's preferred language can be added by them since keyboards have different language options. This is when our Firebase Analytics come into play where we can see where most of our users are located. A good example of this would be of a user who is living in Japan and has an Android phone with Japanese keyboard. We can change our default labels into Japanese so that the users can read the default labels. With the Japanese keyboard, the user can add new labels which won't crash the application.<sup>[7]</sup>

## References

- [1]"Gitflow Workflow | Atlassian Git Tutorial", Atlassian, 2018. [Online]. Available: <https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow>. [Accessed: 03- Aug- 2018]
- [2]"Mobile App Testing on Devices – AWS Device Farm", Amazon Web Services, Inc., 2018. [Online]. Available: <https://aws.amazon.com/device-farm/>. [Accessed: 03- Aug- 2018]
- [3]"Upload your App - Diawi - Development and In-house Apps Wireless Installation", Diawi.com, 2018. [Online]. Available: <https://www.diawi.com/>. [Accessed: 03- Aug- 2018]
- [4]"Jenkins User Documentation", Jenkins User Documentation, 2018. [Online]. Available: <https://jenkins.io/doc/>. [Accessed: 03- Aug- 2018]
- [5]"Firebase", Firebase, 2018. [Online]. Available: <https://firebase.google.com/products/>. [Accessed: 03- Aug- 2018]
- [6]"Google AdMob - Mobile App Monetization & In App Advertising", Google.com, 2018. [Online]. Available: <https://www.google.com/admob/>. [Accessed: 03- Aug- 2018]
- [7] M. Perroni, "How Can Configurable Business Logic Improve Labeling?", Loftware.com, 2018. [Online]. Available: <https://www.loftware.com/blog/how-can-configurable-business-logic-improve-responsiveness-for-labeling>. [Accessed: 03- Aug- 2018].