BSC APPRENTICESHIP DEGREE

DIGITAL TECHNOLOGY SOLUTIONS (DTS)

LEVEL 6

Final year Portfolio

BSc Apprenticeship Degree in Digital & Technology Solutions

Software Engineer Pathway

Veronica Sonzini

Cache of UI events during Bootstrap

Task title	**Cache of UI events during bootstrap**
Apprentice role	Software engineer
Company's name	Google
Date of submission	18th of December, 2017

Introduction & Background

I'm Veronica Sonzini and I'm a Software Engineer Apprentice at Google. I'm currently in the first year of my apprenticeship program.
Google is a very big company that has a worldwide presence and focuses on may different products such as:

- Google search engine (main product and most famous)
- Android operating system for mobile devices
- Chrome web browser, and other products

I'm a part of a team that built a framework for building very lightweight UI components that are used by other teams in Google. My team is part of a bigger business area called Android Google Search App, that deals with the development of everything included on Google Search app for Android devices.

"M" framework

"M" is a framework for the Android Google Search App that provides an extension point for building units of UI called "*M features*". Each feature is split into a renderer and a controller.

Renderers live in the Android Google Search App’s UI process, and are meant to be lightweight, responsive and easy to bring up or tear down when Android reclaims memory.

The controllers live in the search service. The search service takes care of most of the heavy work such as networking and other I/O.

Bootstrapping

“M” framework’s bootstrap process creates the potential for renderers to be initialized before the corresponding controllers. This is not ideal, especially when the search service is not up at the same time as the UI (renderers), because it means the user sees an empty view until the search service is available and has created a controller, and can retrieve the data to populate the view.

Under bootstrapping the UI side can start loading and instantiating renderers before it hears back from the service side. The result is that the user sees things being loaded sooner than they would otherwise.

UI events & Why they can be lost

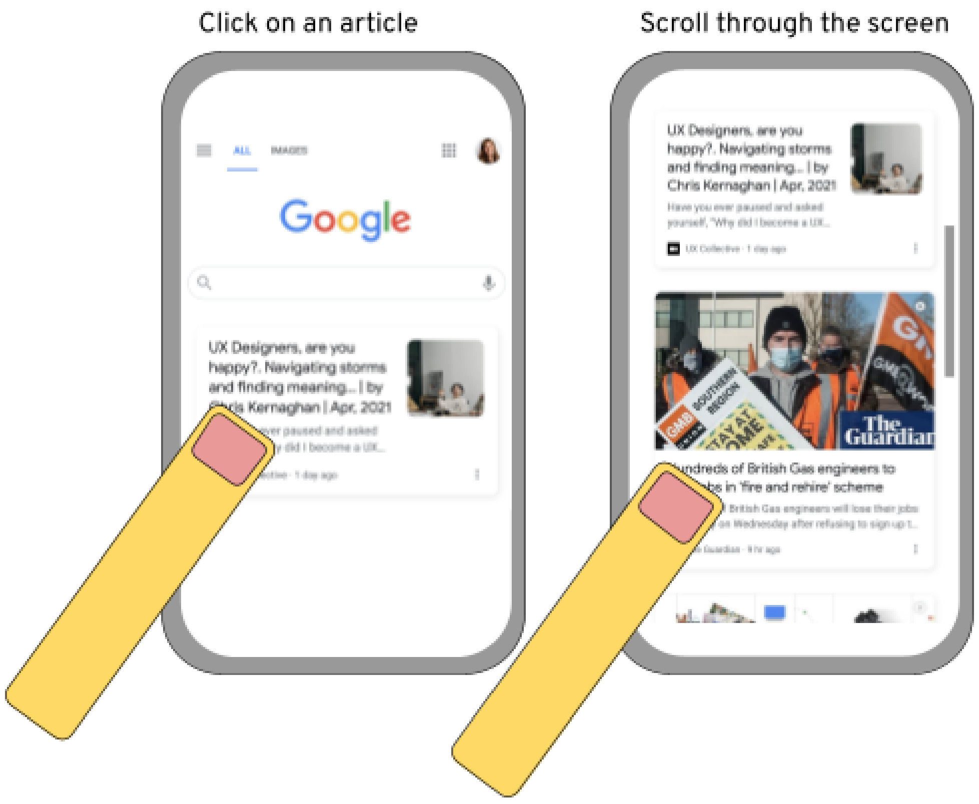
What is UI?

At the most basic level, the user interface (UI) is the series of screens, pages, and visual elements—like buttons and icons—that enable a person to interact with a product or service. In this case we’ll be referring to mobiles specifically.

UI events can be any action triggered by the user from their phone:

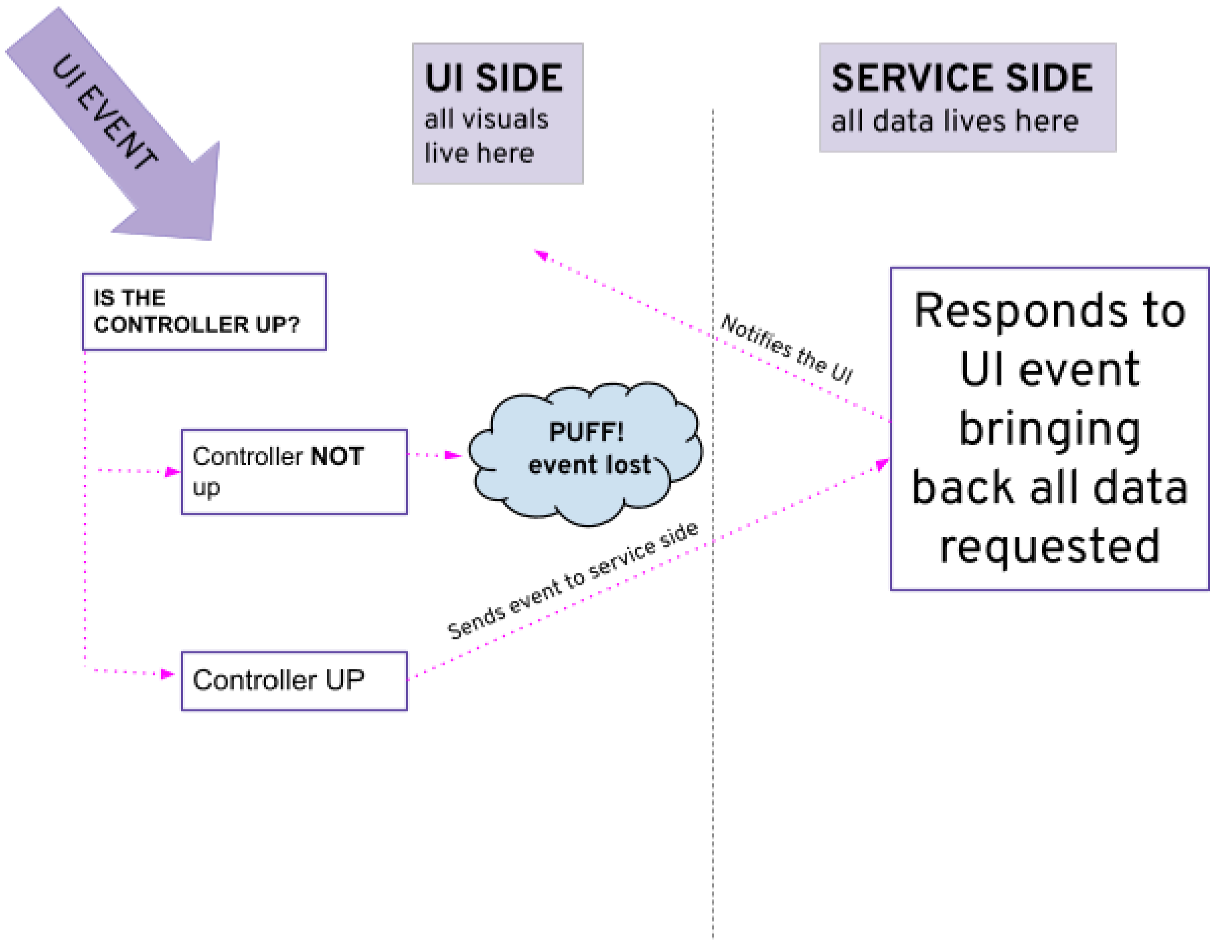
- The user clicking on an article on Google app
- The user scrolling the screen
- The user clicking on the menu

When the user interacts with Google app, they expect something to happen right away. If I click on an article, I expect to see something visual that indicates to me that article is loading (loading bar, spinner, etc) and then I expect to see the article on the screen.


Click on an article
Scroll through the screen
Google
UX Designers, are you happy?. Navigating storms and finding meaning... | by Chris Kernaghan | Apr, 2021
The Guardian

How can UI events get lost?

When an event is triggered (the user clicked on something), this is captured by the UI (since whatever the user interacted with is a UI element). It could happen that while the UI side is up, the service side is not yet up, so then there's nothing to "catch" this action the user is performing, so the event is lost. This could result in the user clicking on a button, for instance, and nothing happening. The user would click and click and no response would be visible. Once the controller is working, it has no way of knowing that the user has done something (clicked a button) and nothing will happen. The user will be very frustrated.



Aims & Objectives

The problem

The problem to solve is how to hold the events while the service side is down, so they don't get lost.
In order to keep these events until the controller appears, a suitable data structure is needed to hold events until the controller is up. The said data structure will need to keep the events in the order they appeared: The first event added to the structure will need to be the first one shipped to the service side once the controller is up (first in, first out).
This structure of events needs to be clearable. If the controller never comes up, the structure should be discarded, and if it does come up, the structure should be sent to the controller and then be discarded.

The solution

If the controller is not yet initialised, we'll need to create a queue to hold the events sent by the user until the controller is back.
The queue will live on the UI side.
We need to add events to the queue and wait until the controller is up. Once it's verified that the controller is up, each event is sent to the service and the queue is cleared.
If the controller never comes up, the queue will need to be cleared.

Objectives

1. Create a queue to receive the events
2. Add a function to clear the queue after a certain time
3. Add a function to dispatch the events in order to the controller
4. Add a function to clear the queue once the events are dispatched and received

Stakeholders

- "M" team: composed of 6 people, including myself:
    - Tech lead
    - 3 senior Software Engineers: one of them was my host in the team
    - Myself (Software Engineer apprentice)
    - Manager
- All internal teams that are part of the Android Google Search App business area: these are our end users

Timescales & Budget constraints

Timescale	**Budget**
September - December 2017	N/A

During this project there were a few constraints related to my available time for work. Throughout the time I've been working on this project I had to be pulled out of work for several reasons, for example:

- Attending internal trainings
- Attending Ada college
- Technical learning sessions such as Java (the coding language I used during my time in this team) training

Methodology

My team worked following an Agile-like approach. We had weekly meetings where we would align and learn what other teammates were doing, and what they were working on. We used this time also to plan what we would be doing during the week. We also had retrospective meetings every few weeks.

During this time I learned a lot about

Design

Implementation can be segmented into four key parts:

1. Linking this new list infrastructure to the existing "M" UI event handling.
2. Sending events to the service and deleting all data from the list once the controller is up.
3. Hold the UI events in case the home button is pressed while the controller is down.
4. Testing

1: Linking queue infrastructure to existing "M" UI event handling

- Create a List as a member of the UI side class.
- Add the appropriate getters.
- When the controller is not up it will create a UI event with the event type, event source and event data available. It will add the UI events to the List.
- The renderer will be notified when the controller is up, and it will send the ui events to the service side.

2: Send UI events to the controller once the controller is up.

- Hold events on the UI side until the controller is initialized. An update receiver method will notify the renderer when the controller is initialized.
- After sending all the events the list is cleared.
- Events will be sent in chronological order.

3: Hold the UI events when SurviveOnStop is true.

The list would get destroyed when the renderer goes away. In the case the Home button is pressed, the renderer is kept alive but the controller gets destroyed. We need to hold the events also in this case until the controller is back up.

4: Testing

- Requires creating an artificial delay in the controller initialisation to test Events are queued and then executed.
- Edge cases
  - Several events of the same type
  - Controller never appears
  - Users ‘leaves’ the feature before bootstrapping is complete (e.g. impatiently closes Google Search app).

Implementation

First I created all corresponding technical documentation explaining step by step what was my approach to this project. I submitted said document for a formal Google Search team design review. This was approved by all the team members and manager.
I created a very detailed design diagram, using an online tool, improving the visualisation of the feature implementation.
I then created a feature flag to guard the code. This was done following the internal feature launching guide.
I was expecting this new functionality to have a very small impact on performance. I measured the impact with an internal tool.
I wrote unit tests to ensure my code was completely functional and wasn’t breaking any of the existing functionality.

All my changes were submitted via an internal tool. Peer feedback on my code was also submitted and addressed in this same tool (similar to how Github is used for this)
I participated in several team meetings where we would align our work, and show my progress.
Feedback was continuous and I had to refactor my code multiple times until it got to the expected way and then I could merge and deploy.

Testing & Verification

Unit tests for all new functionalities: Total test coverage ranged between 60 - 100%

Tools

The software tools used in this project:

- Android Studio IDE
- Internal CI (continuous integration)
- Terminal
- Internal diagramming tool

Evidence

Skills	Evidence
Skills: 1. Create effective and secure software solutions using contemporary software development languages to deliver the full range of functional and non-functional requirements using relevant development methodologies. 2. Undertake analysis and design to create artefacts, such as use cases to produce robust software designs. 3. Test code to ensure that the	**Feedback** **Vera D - Host Feedback** **Googler's Impact: Critical** This is the first self-contained project Veronica was working on after a few starter projects. It required learning a lot of new things: new UI framework, bootstrapping principles, data structures and more. This project also implied communication with many new people from the team. Veronica handled all these complexities well and with some help of teammates she designed the

functional and non-functional requirements have been met. **Technical knowledge:** 1. How to apply software analysis and design approaches. 2. How to interpret and implement a design, compliant with functional, non-functional and security requirements. 3. How to perform functional and unit testing. 4. How to use and apply the range of software tools used in Software engineering. **Behavioural skills:** 1. Fluent in written communications and able to articulate complex issues. 2. Makes concise, engaging and well-structured verbal presentations, arguments and explanations. 3. Flexible actitud. 4. Ability to perform under pressure. 5. Logical thinking and creative approach to problem solving.	flow for cache UI events whilst bootstrapping. The work was split into a few steps: ● Writing design doc ● Prototyping the full functionality ● Breaking prototype into smaller changes ● Producing code ● Adding tests The first two phases required a lot of effort in learning many new things (framework, bootstrapping principles, data structures etc). Veronica has had the tenacity to go through these phases successfully. The third phase involved deep thinking of the code structure and creating thorough tests for the new functionality. There have been many roadblocks along the way, and Veronica found all needed workarounds via investigating issues on her own or getting help from teammates. For example, Veronica synced with a Senior Engineer of the team, who gave her overview of unit-testing gotchas. Also, Veronica was very open to getting and incorporating all the feedback. **Philipp S - Tech Lead - co-worker Feedback** **Googler's Impact: Critical** The task was quite complex. Veronica needed to have a basic understanding of

	the whole UI framework to implement it. The feature is to queue user interactions until the feature is completely initialized during bootstrapping, which lets us avoid any product excellence issues where user interactions would get lost if they happen too early. Veronica successfully implemented the change with great unit test coverage, and it did not cause any problems. Having this implemented means we can continue with our plans of bootstrapping search suggestions, which will reduce the suggestion latency by more than 600ms. **Eugene G - Feedback from Manager** **Googler's Impact: Large** Veronica participated in the design discussions and co-authored the design doc. Learned a lot about the new UI Framework and a few new programming concepts. Started coding a prototype of this work.

Conclusion

Outcomes

All goals have been achieved with the solution I proposed and implemented. This was reflected in the high impact results of the work:

- Great reduction in latency (the exact percentage can't be shared)
- High increase in Google queries (exact number can't be shared)

- High increase in Google Search Daily Active Users

Bootstrapping has been a success. This was a great cross functional effort.

Insight Opinion

What did you learn from key learning experiences?
Because of this project I learnt how valuable cross team collaboration is. It was an effort of many different teams all working with the same objective: get bootstrapping working to reduce latency, get more people to stay using Google search on their devices, and ultimately give a great experience to users. As my first real project I learnt how to communicate with people from my team, and from other teams as well. I also learnt how to conduct myself in meetings, how to ask questions and to whom. This was a very valuable first project.
How have you grown because of these experiences?
I have grown as a developer and as a team member. This project definitely amplified my technical skills, gave me practice in understanding how the whole company's development system worked. I learnt how to use many of the internal tools available for development, from continuous integration, to filing bugs and receiving feedback.
Is there anything you might do differently next time?If I had to do this project again,
I'm not sure I would change anything, since the whole process was very smooth. I got a lot of assistance from more senior team members who were always happy to give feedback and assist me if necessary. It would have been helpful to know the tools beforehand, but this was impossible since it was my first project in the company.

Analysis

What were key factors that shaped this experience?
For the first time as an apprentice at Google I had to come up with a solution myself, and even though it was very scary (as part of Google, I didn't think I was up to the task at the point) I proved to the team and myself that I could do the task.
Having to go through multiple layers of feedback, debugging and refactor was hard, but this led me to success.
How has this experience changed you as an individual?
Now, as an individual, I know how to communicate better, at what point I require assistance from other team members. My self esteem improved a lot from the moment I saw my changes being submitted to production. I even received an award from a team member thanking me for all the hard work.
What factors could have changed the outcome of this experience?

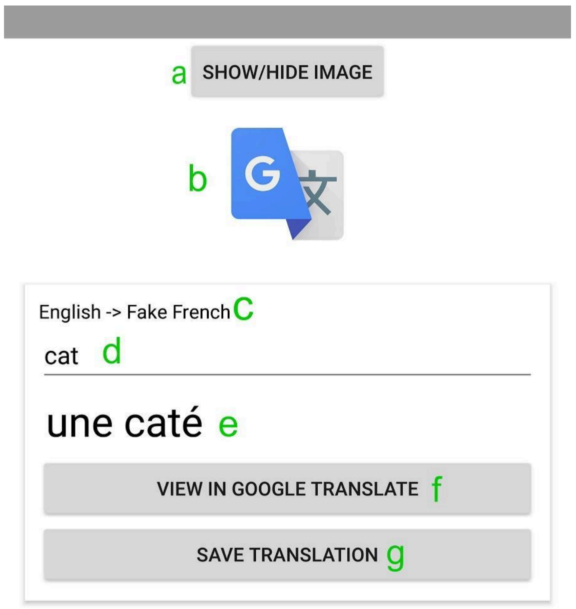
In my opinion the time frame given for this project was enough, it gave me time to understand the problem, research the documentation about bootstrapping and all the work that had been previously done. If the time frame had been smaller, or the team would have needed a solution right away, I don't think the project would have been as successful from my part. So I would definitely say that the time factor would have changed the outcome of this experience/

UI tests for "Translate demo"

Task title	**UI tests for Translate demo**
Apprentice role	Software engineer
Company's name	Google
Date of submission	21st of March, 2018

Introduction & Background

"M", as explained in the previous project's background, is a framework for the Android Google Search App that provides an extension point for building units of UI called "*M features*". The UI created using this framework is super lightweight.
In order to promote the use of the "M" framework, demo projects were created using the lightweight UI. The "Translate demo" is the first demo created using "M". This demo was a language translator, and for the sake of the demo concept, the only language it translated to was an invented type of French: the word "cat" would translate to the word "caté" (which is definitely not the french for cat!).



Aims & Objectives

UI tests are tests that verify the UI is correct, and that the flow between components is what is supposed to be. This type functional tests help check that any new feature introduced to the app, or changes, are not altering the expected UI. This project was all about creating UI tests to cover all possible scenarios of the app usage.

The objectives of this project were divided into 3 stages:

1. Create a detailed design document describing all the UI test cases that were going to be implemented.
2. Creation of the actual UI tests that would be run during the presubmit process. The tests would need to simulate real user flows and verify various aspects of the UI.
3. Add the tests to be run during the pre-submit process (before anyone can submit any code to the translate demo project).

Stakeholders

- “M” team: composed of 6 people, including myself:
    - Tech lead
    - 3 senior Software Engineers: one of them was my host in the team
    - Myself (Software Engineer apprentice)

- Manager

Timescales & Budget constraints

Timescale	Budget
January - March 2018	N/A

During this project there were a few constraints related to my available time for work. Besides the usual training and courses I needed to attend, I also took personal time off. This didn't affect my work nor extended the timeframe in which I completed the project.

Methodology

Design

The tests were divided into 5 groups:

1. **Tests for the initialization of the app**
   Verifies the app loaded correctly and the visibility of UI elements (rendering and viewing of elements)

2. **Tests for the main flow of the app**
   Verifies the demo's main flow: a translation is made, saved and opened on Google Translate.

3. **Tests for restoring the views after a device rotation**
   This test verifies that the views can be restored correctly after rotation.

4. **Tests for restoring the app from "recents"**
   This test verifies that the views can be restored correctly after reopening the app from recents apps after pressing the "Home" button.

5. **Test for restoring a translation in back stack after "return" button pressed**
   This test will verify that after a translation is added to the back stack by pressing the "push to back stack" button, if the back button is pressed, it will restore all translations added to the back stack, last in first out.

Implementation

To start this project I first had to create a design document with all the details of the case scenarios I would be covering. I had to make sure I was covering all obvious scenarios and edge cases.
I then had to create tickets for each of the test cases.
Writing the tests was the most lengthy part of the project, since after writing them they had to be run to verify they were actually working. To run each test would take several minutes, and usually I had to run each test multiple times until I had it working the way I wanted.
To test UI I had to assign labels to each of the UI elements on the screen:

- “translate_button”
- “save_button”
- etc

And I had to see if my test would match whatever was happening on the UI. For this I had to create static methods.

Testing & Verification

This was explained already in the “Design” section of this project.

Tools

The software tools used in this project were:

- Google docs for the creation of the design document
- UI automator for UI tests
- Android Studio
- Java coding language
- Continuous integration internal tool

Evidence

Skills	**Evidence**
Skills:	**Feedback:**

1. Create effective and secure software solutions using contemporary software development languages to deliver the full range of functional and non-functional requirements using relevant development methodologies. 2. Produce high quality code with sound syntax in at least one language following best practices and standards. 3. Perform code reviews, debugging and refactoring to improve code quality and efficiency. 4. Test code to ensure that the functional and non-functional requirements have been met. 5. Undertake analysis and design to create artefacts, such as use cases to produce robust software designs. **Technical knowledge:** 1. How to perform functional and unit testing. 2. How to use and apply the range of software tools used in Software engineering. **Behavioural skills:** 1. A thorough approach to work. 2. Logical thinking and creative approach to problem solving. 3. Able to conduct effective research, using literature and other media, into IT and business related topics.	**Vera D - Host Feedback** UI tests are a big pain for engineers (used emulators are often flaky, testing API do not always consider all corner cases). For this project, though being familiar with the framework's infrastructure, Veronica again had to learn a lot of new things: general UI tests structure, testing framework, continuous build workflow. **Philipp S - Tech lead / co-worker Feedback** The translation demo is a demo of the UI framework. Before Veronica's work, this feature had no UI tests, and developers accidentally broke it multiple times, and often the demo was broken because the framework was actually broken. It usually took us weeks to notice this. Veronica implemented a thorough UI test coverage for the translate feature, which means that now whenever we break a core framework functionality we notice this quickly as the translate demo test is failing. When I added a back stack to the demo after the main test coverage was there, Veronica suggested adding a UI test for the back stack functionality too. She added the test, which is great because it means the back stack is one of the first team's features that has UI test coverage from the start. **Nohemi F - Host Feedback** Very complete design doc outlining the cases that needed to be covered by the

	subsequent work. Things that are working well ● Takes initiative in refactoring code ● Understands overall goals of implementation Improvements that can be made across code reviews *Comment*: the improvements are mostly in the details of a code change ● Better understanding of BUILD files (rules + visibility) ● Better understanding of what is being changed and why ● Better understanding of code reusability / unused (dead) code

Conclusion

The test cases were successfully created and this meant that the demo wouldn't be broken by accident, since all tests were automatically run before any piece of code was submitted to the demo app project.

Outcomes

Have the goals been achieved? Why?
All goals were achieved, all test scenarios were covered with test cases.

Insight Opinion

What did you learn from key learning experiences?
I learnt about end to end UI tests, the value of having them, and how to add them to the automatic system of pre-submit.
How have you grown because of these experiences?
This was the first project I worked on completely by myself. In previous projects I always had to work with other team members, but this time I had to do it all. This meant I had to organize

the timeline to work on this, do all the design myself, etc. It was a very rich opportunity and it helped me learn how to manage my own time.

Is there anything you might do differently next time?

Now that I'm a lot more familiar with the UI testing environment, I guess that next time I encounter a similar task I would do it in less time and will have less "unknowns".

Analysis

What were key factors that shaped this experience?

Having to work alone in this project definitely shaped my experience working. It had positives and negatives.

How has this experience changed you as an individual?

I had to learn how to ask for help from people that weren't involved with this project. This helped me overcome the fear of "asking dumb questions" for sure.

What factors could have changed the outcome of this experience?

When I was on board for this project, my host really took the time to explain it all thoroughly. This was overall a very positive work experience. Not having a person to explain to me what the project was about, and how the demo worked, etc, would have make this experience less enjoyable.

Team's introduction and advanced guide

Task title	**Team's introduction and advanced guide update**
Apprentice role	Software engineer
Company's name	Google
Date of submission	14th of May, 2018

Introduction & Background

Many teams at Google have an introduction guide or tutorial with working best-practice example code, and code exercise to help beginners get up to speed with key Google technologies.
There are around a hundred codelabs at Google (perhaps more, I didn’t count them all).
The “M” team is no exception, and they have an introduction and an advanced guide, where it explains step by step how to create something using the “M” framework.
The guide or tutorial is written by team members and because it’s open to anyone at Google, people can leave feedback on it, create bugs if something it’s not working as expected, etc, until it reaches a point where the tutorial needs to be updated and bugs fixed.

There were 2 tutorials:

1. **Introduction guide**
   “M” introductory tutorial would help you get started on your own “M” feature by building a simple static plugin for Android Google Search App. In this guide you would learn how to:
   - Split your “M” feature into a controller, renderer and model
   - Add a new “M” static plugin to Android Google Search App
   - Use the model and UI events for communication between UI and service side
   - Define an interface shareable with other plugins

2. **Advanced guide**
   This tutorial would help you create an image recognition feature to display images that trigger image recognition logic on click and render the result.

Aims & Objectives

The objective of this project was for me to update all tutorials:

- Address feedback
- Fix bugs
- Update content that was out of date.

Stakeholders

- “M” team: composed of 6 people, including myself:

    - Tech lead
    - 3 senior Software Engineers: one of them was my host in the team
    - Myself (Software Engineer apprentice)
    - Manager
- People external to my team who need to learn how to use our framework by using these tutorials.

Timescales & Budget constraints

Timescale	**Budget**
April - May 2018	N/A

Methodology

Design

The design on this project was split into different stages:

Introduction guide

1. Address comments
2. Fix bugs
3. Refactor code: remove legacy code, replace with up to date code
4. Clarify & update concepts
5. Continuous testing from beginning to end
6. Submit changes

Advanced guide

7. Address comments
8. Fix bugs
9. Refactor code: remove legacy code, replace with up to date code
10. Clarify & update concepts
11. Continuous testing from beginning to end
12. Submit changes

Implementation

I started by reading and going through the guides, trying out each exercise. After I did all the research necessary, I dived into addressing all comments left by users:

- Some concepts were not clear enough
- Some code that was old, wasn't working as expected
- Grammar errors found

After carefully understanding what was needed (from the user's point of view) I took over all technical requirements. This involved:

- Fixing bugs in the code
- Importing new libraries that were important
- Updating or completely removing code methods
- Adding new methods to the code
- Updating interfaces
- Removing unnecessary dependencies

This stage involved testing: unit tests and manually testing each exercise myself.

After updating all code and technical requirements I dived into clarifying concepts:

- Modified the explanations to make them more clear and understandable
- Added new examples
- Added images and graphics
- Addressed grammar errors.

Testing & Verification

Testing was done continuously in every step. This involved unit testing for the code, and manual end to end testing performed by me and my team mates. I asked them to do the tutorials from zero, and I collected the feedback. I had to reiterate on many of the concepts.

Tools

The tools used in this project:

- Android Studio IDE
- Google drawing

Evidence

Skills	Evidence
Skills: 1. Contemporary techniques for design, developing, testing, correcting, deploying and documenting software systems from specifications, using agreed standards and tools. 2. How to deliver a technology solutions project accurately consistent with business needs. **Technical knowledge:** 1. How teams work effectively to produce technology solutions. 2. Undertake analysis and design to create artefacts, such as use cases to produce robust software designs. 3. How teams work effectively to develop software solutions embracing agile and other development approaches. 4. How to apply software analysis and design approaches. **Behavioural skills:** **1.** Fluent in written communications and able to articulate complex issues. 2. Makes concise, engaging and well-structured verbal presentations, arguments and explanations.	**Feedback:** **Pragya B -Host Feedback** **Impact: Medium** The Monet codelabs are very useful and detailed for feature developers to start using the core “M” framework in their features.

3. Able to conduct effective research, using literature and other media, into IT and business related topics. 4. A thorough approach to work. 5. Logical thinking and creative approach to problem solving.	

Conclusion

Outcomes

Have the goals been achieved? Why?
The tutorials were successfully updated, the bug's list reduced to zero, and the feedback was very positive.

Insight Opinion

What did you learn from key learning experiences?
This project gave me the opportunity to deeply understand how our framework worked. I had to go beyond what I initially knew and deep dive into the concepts.
How have you grown because of these experiences?
Working in teams is extremely important for developers, as the value is greater. But also, it is very important to know how to work alone, how to self manage the time, keep finding things to work on when I'm finished with a task, basically self-manage. This was a great opportunity for all this.
Is there anything you might do differently next time?
I involved my teammates late in the process, and this led to a lot of reiteration on some bits of the project: like explaining concepts in a different way. If I had involved other people from the beginning I might have completed the project in fewer days.

Analysis

What were key factors that shaped this experience?
This project wasn't an entirely "engineering" one, since it involved a lot of explanation, creation of images and graphics to better explain concepts. It was very entertaining and refreshing to move out of "coding" and do something a bit different.

I worked by myself on this, and it was very good because I could organize my work the way I thought it was more convenient.
What factors could have changed the outcome of this experience?
Earlier feedback from my peers would have made a huge difference, since I ended up spending many days changing things I had already changed. I had to go through many iterations with some of the concepts explained in the guide until I had the “thumbs-up” to make it public.

Voice - languages migration

Task title	**Language settings migration**
Apprentice role	Software engineer
Company’s name	Google
Date of submission	16th of August, 2018

Introduction & Background

This project involved adapting the existing code from the “Language” settings in Settings app (Android devices) to work with the new design my team had developed.
The intention of the change in the architecture design of the Language settings was to simplify the written code, making it simpler to modify and to extend.
This involved new design documentation for the architecture of the Language settings feature that was changed, as well as various meetings with the team to discuss the best approach to achieve the objective.

As previously stated, my team created a Framework for UI features in the Android Google Search App. By using our UI framework, feature developers can benefit from many performance improvements when using it, like:

- Developers can focus on the unique requirements of their application instead of spending time on application infrastructure (plumbing).

This UI framework can be used to create UI features, following a specific Design Pattern that makes these elements lightweight (adaptive and flexible).

Right now whenever a team needs an adapter for RecyclerViews, they would create a custom adapter or use the recyclerView adapter designed by the “M” team. This is a new, simpler recycling Adapter that’s meant to integrate well with existing features with a simpler and less invasive solution.
Using “M’s” new recycling Adapter will save Feature teams the trouble of creating their own Adapter any time they need to display a vertical scrolling list.
Resolving one team’s issue will impact on all the other teams that want to use our platform as well.

Aims & Objectives

The main objective of my project was to simplify the code base of another team, by replacing an Android component that they are currently using on their feature, with a simpler version:

1. Create a list of languages in Android’s Settings app, replacing their custom - single use - RecyclerView adapter by our generic and reusable RecyclerView adapter.
2. These text items will display the language name and a checkbox for language selection, divided into two different sections:
    - Suggested Languages
    - All Languages

The biggest challenge was to adapt the existing code to use our design patterns. This was still a work in progress, it involved a lot of refactoring and a complete overhaul of the existing design.

Stakeholders

1. “M” team: composed of 6 people, including myself:
    a. Tech lead
    b. 3 senior Software Engineers: one of them was my host in the team
    c. Myself (Software Engineer apprentice)
    d. Manager
2. The team I’m a part of, by using this as a proof of concept, where the infrastructure built by my team was applied to another team’s feature. This can be thought of as internal “advertising” for feature developers to see how our UI framework works.

3. The team where the feature is going to be implemented, considering that their code base is being simplified and reduced: less code and less to maintain.

Timescales & Budget constraints

Timescale	Budget
No hard deadline. Completed May - July 2018	N/A

During the time I worked on this project, unfortunately I had an accident while commuting to work, and had to take time off for medical reasons. After not being able to work for a week, I resumed my work with the project. Despite this incident, the project was finished on time.

Methodology

Design

It took me over one week including team planning, and various meetings to get to the final design.

I took an adaptive staging approach for this project, rather than a total waterfall

approach or agile, dividing the process (in an informal way) into phases, stages and decision points.

The process began with a framework (design) that could be modified, if necessary, by new information.

Decision points marked the transition between stages of the project's implementation. At these points, an evaluation of the results obtained was made and the path to proceed was decided.

This approach was all about continuous learning throughout project development: integration of new knowledge.

Similarly, decision points allowed me to adapt and incorporate new information throughout the process. Stages were defined to pursue continuous improvements until success was reached.

Implementation

The project was implemented on the Android platform, following OOP.

Implementation was split into different steps:

1. Creating all the Models.
2. Creating the Views.
3. Adapting the existing Controller to adopt the new design.
4. Splitting the current Events Interface.
5. Replace the RecyclerView Adapter.

Testing & Verification

Tools

The software tools used in this project:

- Android Studio IDE
- Internal CI (continuous integration)
- Terminal

Evidence

Skills	**Evidence**
Skills: 1. Create effective and secure software solutions using contemporary software development languages to deliver the full range of functional and non-functional requirements using relevant development methodologies.	**Feedback:** **Pragya B - Host Feedback** **Impact: Critical** Veronica was the first engineer to use the infrastructure support for recycling in the Voice languages plugin. Some parts of the migration to the new infrastructure were quite complex and required technical design

2. Undertake analysis and design to create artefacts, such as use cases to produce robust software designs. 3. Produce high quality code with sound syntax in at least one language following best practices and standards. 4. Perform code reviews, debugging and refactoring to improve code quality and efficiency. 5. Test code to ensure that the functional and non-functional requirements have been met. 6. Deliver software solutions using industry standard build processes, and tools for configuration management, version control and software build, release and deployment into enterprise environments. **Technical knowledge:** 1. How to operate at all stages of the software development lifecycle. 2. How teams work effectively to develop software solutions embracing agile and other development approaches. 3. How to apply software analysis and design approaches. 4. How to interpret and implement a design, compliant with functional, non-functional and security requirements. 5. How to perform functional and unit testing.	thinking to refactor the code. Veronica took the initiative to collaborate with different team members of the team to design the feature along with writing a design doc on it.

6. How to use and apply the range of software tools used in Software engineering. **Behavioural skills:** 1. Fluent in written communications and able to articulate complex issues. 2. Able to deal with different, competing interests within and outside the organisation with excellent negotiation skills. 3. Competent in active listening and in leading, influencing and persuading others. 4. Able to give and receive feedback constructively and incorporate it into his/her own development and life-long learning. 5. Able to put forward, demonstrate value and gain commitment to a moderately complex technology-oriented solution, demonstrating understanding of business needs, using open questions and summarising skills and basic negotiating skills.	

Conclusion

Outcomes

Have the goals been achieved? Why?

This project presented a new option for feature developers when it comes to using a RecyclerView. It was proven how this new and generic Adapter for RecyclerViews would reduce the time and number of lines of code that the developer would need to invest when working on this task. It was shown how this new Adapter can be applied and used in multiple - various projects. It was stated the use of this generic instance adapts to any type of project, minimizing the interference with the original code.

Insight Opinion

What did you learn from key learning experiences?
I learnt a lot about views, recycling views and code migration during this project. It was, so far, the biggest project I ever worked on since starting at Google.
I had to learn how to take the biggest problem to fix, split it into smaller problems that I could solve, and make it all work in the specified time frame.
How have you grown because of these experiences?
This was by far the most stressing time during my second year at the apprenticeship program. I had to learn how to manage my expectations regarding the results of the project, and iterate constantly.
Is there anything you might do differently next time?
Next time I would ask for more help from more senior engineers, rather than taking too long to understand a certain concept. I was trying to tackle all problems by myself, to demonstrate I could manage a project this big, but instead of working in my favour this set me back and made me lose valuable time.

Analysis

What were key factors that shaped this experience?
This project started very differently than other projects. Rather than had been given a task to work on, I had to find the project myself, and with help from my host we had to convince another team to let me migrate their current code to use our “M” framework. This wasn’t too hard to do, but definitely was a new experience for me. I had to “sell” to this team my idea.
How has this experience changed you as an individual?
This was the biggest and most complex project I have worked on until that moment. It made me grow as a developer, as I had to solve a lot of issues that I didn’t have to do before. Also, it was my first project I was leading, and this brought a lot of responsibilities.
What factors could have changed the outcome of this experience?

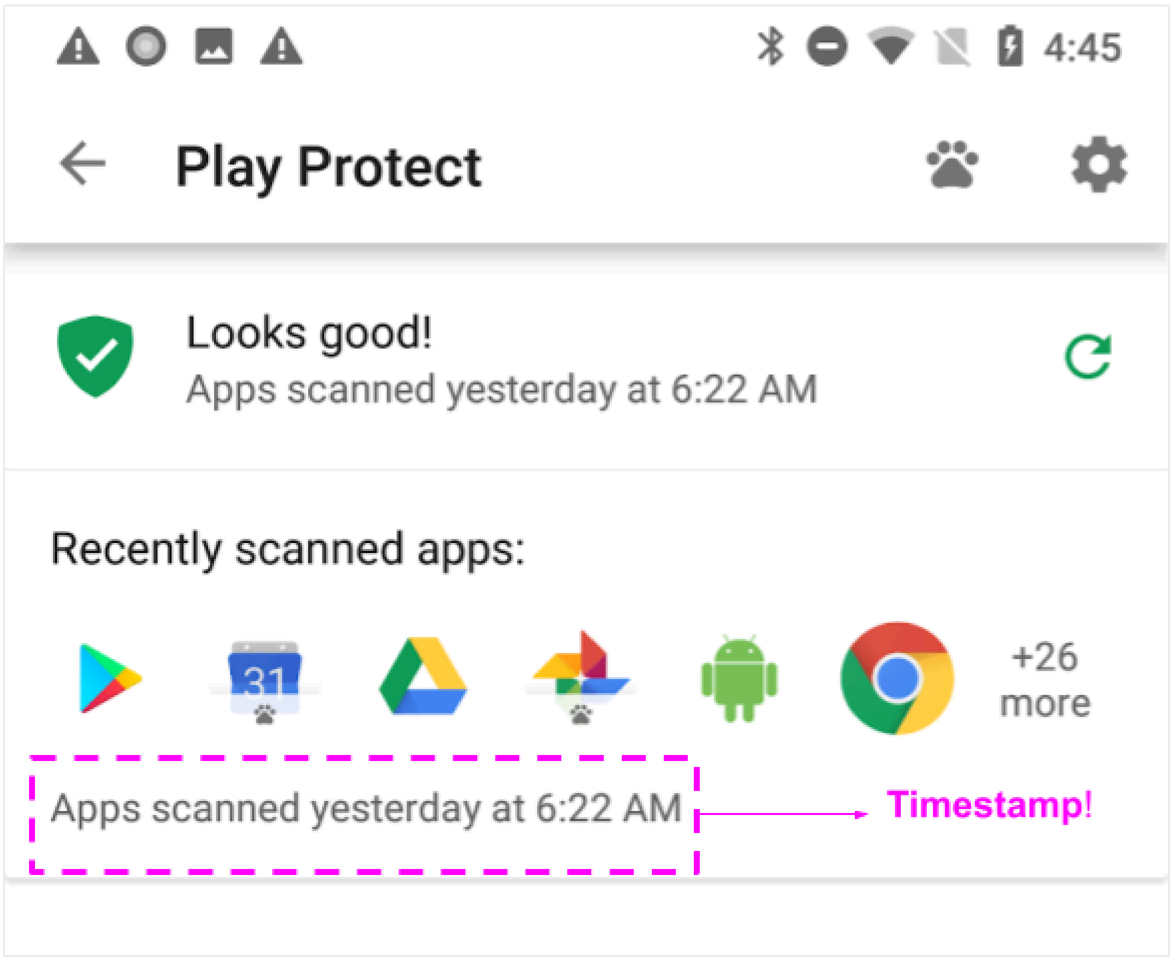
The experience would have been different if instead of leading this project I would have been a collaborator. That would have taken a lot of the stress out of the experience, but at the same time I wouldn't have learnt as much as I did.

Date formatter API

Task title	**Date formatter API**
Apprentice role	Software engineer | UXE engineer
Company's name	Google
Date of submission	21st of February, 2019

Introduction & Background

My project was super simple: I needed to add a timestamp on the newly designed "scanned apps" card for Play Protect.
The "Scanned Apps" card from Play Protect app is a card that displays the installed applications on the user's phone that were recently scanned by Play Protect. This card, besides showing the list of apps, displays a timestamp from the last scan.
As I was working on this really simple, uncomplicated task, I realised the DateFormatter class used on the old "scanned apps" was not clean at all. The code was pretty un-reusable, since most things were hardcoded. This is when I decided we needed a new DateFormatter API.


4:45

Play Protect

Looks good!

Apps scanned yesterday at 6:22 AM

Recently scanned apps:

+26 more

Apps scanned yesterday at 6:22 AM

As mentioned before, the code for generating this timestamp was not reusable and was tightly coupled to the “recently scanned apps” class, as it was built specifically for this use case in particular. Reformatting this code to make it reusable would have meant a great refactoring effort.
I created a new abstraction, created a new completely generalised date formatting API, making it simpler to use, cleaner and reusable.

What is already available in Java?

Java Time library

Can give the date and time "2007-12-03T10:15:30", but it can’t represent day - 1 as “Yesterday”.

RelativeDateTimeFormatter

This is an API used by other projects within Google, and it’s recommended for building more complex APIs.

The solution is to use **RelativeDateTimeFormatter** to create an API that will calculate the time difference between a timestamp of date in the past and the current date/time timestamp, and convert it into a human readable String.
This will return the date in the following format: for example using the long 1540857600000 (30/10/2018), considering the current date 1542153600000 (14/11/2018) it would return "**15 days ago"** or **"October 30th 2018"**, depending on the use case.

Caveats

Date and Time APIs

Currently Java SE has two separate Date and Time APIs -Java.util.Date and java.util.Calendar. Both APIs are consistently described as difficult to use by Java developers. Both use a zero-index for months, which seems to be a cause for many bugs. Calendar also has many bugs and performance issues, primarily due to storing its state in two different ways internally.

Using some of Java 8's newest magic!

Java LocalDate, a completely new API introduced in Java 8, it's loosely based on the Java library JodaTime . This new API dramatically simplified date and time processing and fixed many shortcomings of the old date library.

RelativeDateTimeFormatter is a very basic API
pros:

- can return the specific day ("last Sunday / Next Tuesday") or the elapsed time ("In 3 days").
- Provides correct plural forms.
- Capitalizes the words it formats.

Internationalization (i18n)

Following Internationalization (i18n) rules for dates and time:

- Using ICU DateFormat class
- Using ULocale

- Using **RelativeDateTimeFormatter** to create the strings (“yesterday, 3 PM”): this would fix problems such as not being able to format the time according to the localization of the user.

Aims & Objectives

Create a reusable API to format Dates, from a timestamp to the present time, and displays it like this:

- “Yesterday at xx:xx”
- That displays the time in both AM/PM and 24hr format
- “x days ago”
- “25th of November, 2019”
- Dates from the future:
    - “Next week”
    - “Tomorrow”
    - etc

Stakeholders

Play Protect team is composed of:

- 3 senior Software engineers
- 1 software engineer
- 2 UXE engineers (including myself)
- 2 UX designers
- 1 intern
- 1 QA & testing specialist

Timescales & Budget constraints

Timescale	**Budget**
No hard deadline. Completed January - February 2019	N/A

This project needed to be completed in 3 weeks, which was a tight deadline.

Methodology

Design

V.0: MVP

The MVP (Version 0) goal: Have a DateFormatter that returns a date span: “Yesterday at 4:05 PM”, “3 days ago”, etc.

V.1: Dates of the Future (in the future)

- Next week
- Next month
- Next year, etc

Interface

- Given a timestamp and a clock to calculate the current date and time, it renders a localized representation of the time span.
    - Converts the specified number of milliseconds ago to a string such as “2 months ago” or “Yesterday at 5:30 PM”.
    - Params:
        - **timeMs** → number of milliseconds since the epoch to a specific moment (can be other than the present).
        - **Clock** → the clock will get the current time and date.
        - **Context** → based on the context it can be inferred the format of the time to be shown (12hs or 24hs format). e.g 03:08 PM or 15:08 as well as the user’s locale.

  Options for time span format:
        - **Now** → Less than a second ago
        - Seconds ago → “15 seconds ago” - Less than a 1 minute ago.
        - Minutes ago → “20 minutes ago” - Less than 1 hour ago.
        - Today→ “Today at 3 PM” or “3 PM” - Less than a day ago.
        - Yesterday → “Yesterday at 3 PM” - Less than 2 days ago.
        - Days ago → “4 days ago” - any date between 2 days ago and a week.

        - Date → “October 30” - more than a week ago, up to a year.
        - Over a year → “October 2017”.

Implementation

- **Create a method that receives a Context, a Long for the time in ms, and a Clock that has the current time, and returns a String**
    - Use the **Clock** to get the current timestamp within the method.
    - Calculate the time elapsed between **currentMs** and **timeMs**.
    - Check if **timeElapsedMs** corresponds to “today, yesterday, daysAgo, or date” to define the formattedString.
    - Set constants to define:
        - ONE_SECOND = 1000
        - ONE_MINUTE = ONE_SECOND * 60
        - ONE_HOUR = ONE_MINUTE * 60
        - ONE_DAY = ONE_HOUR * 24
        - ONE_WEEK = ONE_DAY * 7 (This way we can specify the time span as “now” if it’s less than a second).
    - Each case will return the corresponding string produced by the **DateFormatter** class.

Will need to create 2 new files:

1. Interface declaring the methods of the API
2. Implementation of the interface

With this API, the old DateFormatter will become redundant, so this will be removed and replaced.

Testing & Verification

To test the API I’m going to:

- Mock a fixed date and time, and use that as “currentDateMs”.
- Will test the different cases of getRelativeDateFromNow()
    - Today, yesterday, days ago, over 11 months ago, and date in the future. By testing these cases I can make sure the method is working as expected.

Tools

The software tools used in this project:

- Android Studio IDE
- Java 8
- Android's RelativeDateTimeFormatter
- Internal CI (continuous integration)
- Terminal

Evidence

Skills	**Evidence**
Skills: 7. Create effective and secure software solutions using contemporary software development languages to deliver the full range of functional and non-functional requirements using relevant development methodologies. 8. Undertake analysis and design to create artefacts, such as use cases to produce robust software designs. 9. Produce high quality code with sound syntax in at least one language following best practices and standards. 10. Perform code reviews, debugging and refactoring to improve code quality and efficiency. 11. Test code to ensure that the functional and non-functional requirements have been met. 12. Deliver software solutions using industry standard build processes,	**Feedback:** **Simon W - Manager Feedback** **Impact: Medium** Veronica has demonstrated design (through research, design docs, and design reviews), code development, and productionization in this project. They continued to maintain SW for the past 4 months after launch, fixing 6 bugs and adding an API for Data Takeout. However, they did not always have opportunities to demonstrate Requirement Analysis, as the project was part of NextNextNextGoogle and well defined before Veronica arrived on the team. Their Unit Testing was great. Veronica's design approach was thorough in a number of ways, and definitely this helped her progress in the project. Veronica used a range of software tools for engineering, including ● internal and external build tools ● version control tools

and tools for configuration management, version control and software build, release and deployment into enterprise environments. **Technical knowledge:** 7. How to operate at all stages of the software development lifecycle. 8. How teams work effectively to develop software solutions embracing agile and other development approaches. 9. How to apply software analysis and design approaches. 10. How to interpret and implement a design, compliant with functional, non-functional and security requirements. 11. How to perform functional and unit testing. 12. How to use and apply the range of software tools used in Software engineering. **Behavioural skills:** 6. Fluent in written communications and able to articulate complex issues. 7. Able to deal with different, competing interests within and outside the organisation with excellent negotiation skills. 8. Competent in active listening and in leading, influencing and persuading others.	● IDEs (Android Studio) ● emulators ● debug logging and stack trace analysis ● layout inspector tools for Android. Veronica created unit and functional tests for her projects. Her tests were normally clearly written, following closely the conventions of the existing tests, and were sufficiently comprehensive to cover the changes. **Raissa DG - Host feedback** We worked with Veronica together as a team to fix bugs for an upcoming deadline. She contributed 9 fixes from a bug list of about 28 issues (calculated from number of change lists submitted from main contributors).

9. Able to give and receive feedback constructively and incorporate it into his/her own development and life-long learning. 10. Able to put forward, demonstrate value and gain commitment to a moderately complex technology-oriented solution, demonstrating understanding of business needs, using open questions and summarising skills and basic negotiating skills.	

Conclusion

Outcomes

The goals of this project were successfully achieved, and the new DateFormatter API worked perfectly. We are now able to use the DateFormatter in other cards of the application without need of refactoring.

Insight Opinion

From this project I learned about creating APIs. Before this I've never attempted to use this technique to improve the quality of my code, and I can say now that this is a great tool for developers.
I've found growth opportunities in every project I work, and this one is no exception. Every new technique I learn or tool I understand how to use, makes me a better developer. I don't think I would do anything differently if I had to re-do this project, I think the development of the API was very smooth.

Analysis

What were key factors that shaped this experience?
This was my first project in a new team. This factor definitely shaped the experience, since I wasn't familiar at the time with how the team operated. Because of my introverted nature, I had to overcome a lot of my personal anxieties and interact a lot with my new teammates in order to succeed on this project.
How has this experience changed you as an individual?
I can say this changed the way I interact with teammates. Now I'm not as reluctant to ask questions about how to do certain things on my code, or to request feedback more often.
What factors could have changed the outcome of this experience?
I think that if this wasn't my first project at the time, I would have completed it in less time than I actually did. But other than that, I don't really know if other factors could have changed the outcome of my experience. I think it was an overall very positive experience.

Creation of interactive prototype using Framer

Task title	**Creation of interactive prototype using Framer**
Apprentice role	UXE engineer
Company's name	Google
Date of submission	30th of September, 2018

Introduction & Background

Project "A" was a redesign of Play Protect that involved migrating Play Protect from one app to another. As I didn't work on this migration, there's no point in giving an extensive description of this particular project. But to have an idea of how big this project was I can share that most of the documents about this migration had this "meme" on them:

I was asked to work as a UXE on this project, and to create prototypes with the latest UI changes. With the help of these prototypes it would be easier for the UX designers to figure out the flows (identify missing user flows) and visualize the main user journeys.

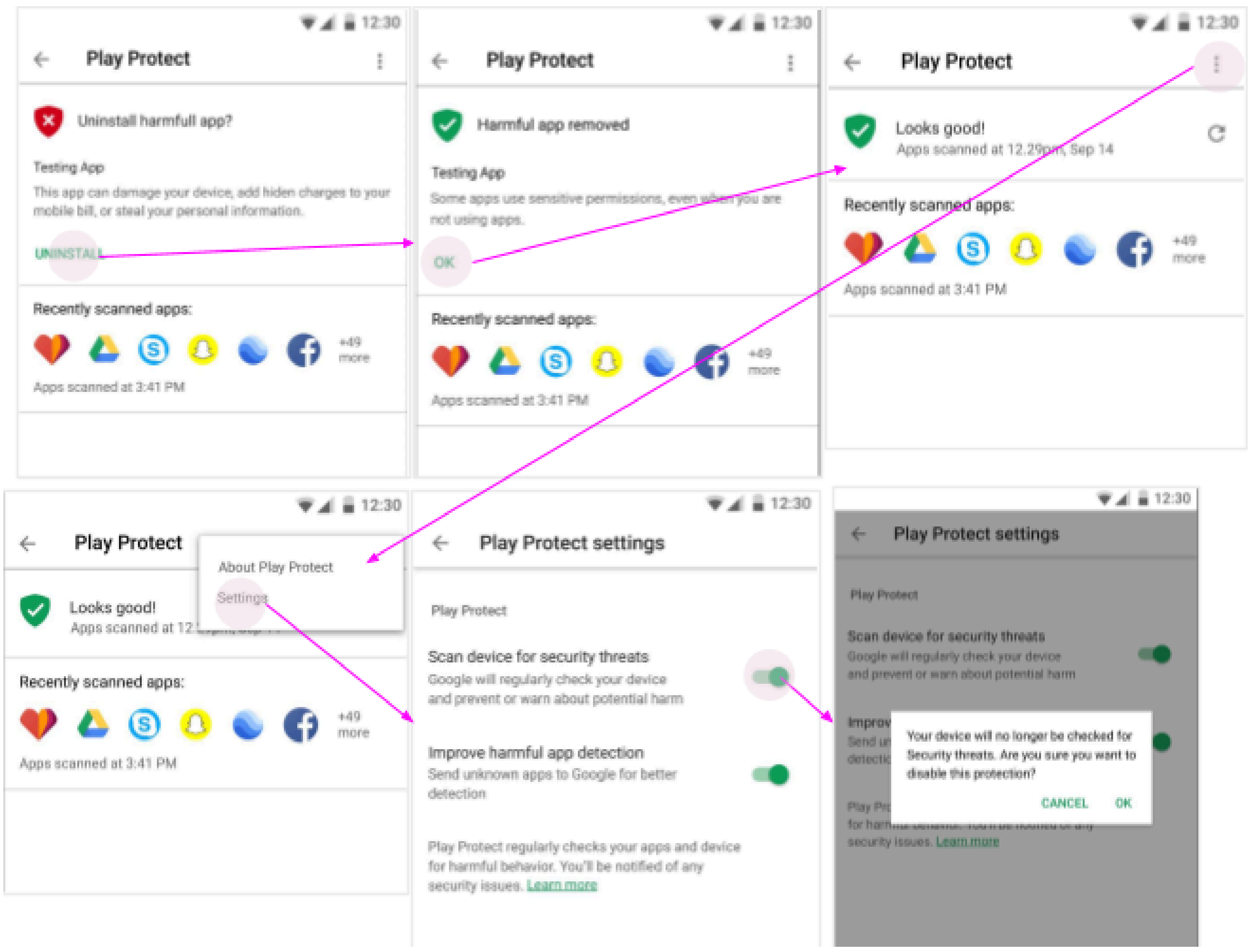
I was issued all the resources needed:

- jpegs
- List of all the entry points to GPP Home
- Moveageddon mocks
- Viewing the prototype on a phone

Framer JS:

Framer is a prototyping tool that creates very realistic interactions. The way you can create prototypes here is by using a coding language called “coffeescript” that compiles into Javascript.

When I was asked to work on this project I had no idea how all this would work, as I was not familiar with this tool in particular or with CoffeeScript language (or javascript for what mattered), but I took the time and made it work.


12:30

Play Protect

Uninstall harmfull app?

Testing App

This app can damage your device, add hiden charges to your mobile bill, or steal your personal information.

UNINSTALL

Recently scanned apps:

+49 more

Apps scanned at 3:41 PM

12:30

Play Protect

Harmful app removed

Testing App

Some apps use sensitive permissions, even when you are not using apps.

OK

Recently scanned apps:

+49 more

Apps scanned at 3:41 PM

12:30

Play Protect

Looks good!

Apps scanned at 12.29pm, Sep 14

Recently scanned apps:

+49 more

Apps scanned at 3:41 PM

12:30

Play Protect

About Play Protect

Settings

Looks good!

Apps scanned at 12

Recently scanned apps:

+49 more

Apps scanned at 3:41 PM

12:30

Play Protect settings

Play Protect

Scan device for security threats

Google will regularly check your device and prevent or warn about potential harm

Improve harmful app detection

Send unknown apps to Google for better detection

Play Protect regularly checks your apps and device for harmful behavior. You'll be notified of any security issues. Learn more

12:30

Play Protect settings

Play Protect

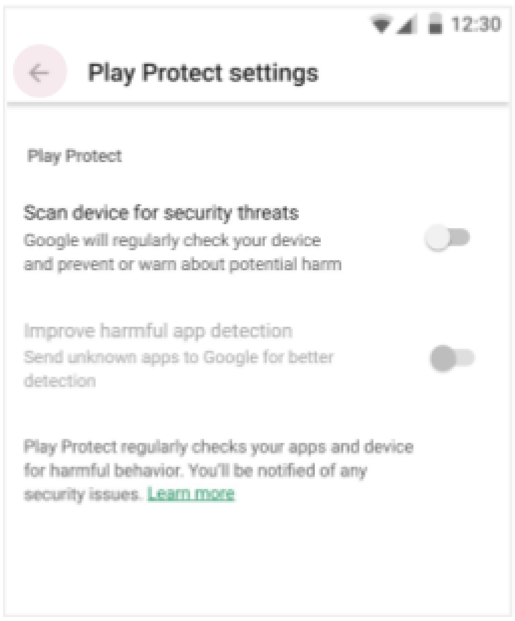
Scan device for security threats

Google will regularly check your device and prevent or warn about potential harm

Your device will no longer be checked for Security threats. Are you sure you want to disable this protection?

CANCEL OK

security issues. Learn more


12:30

Play Protect settings

Play Protect

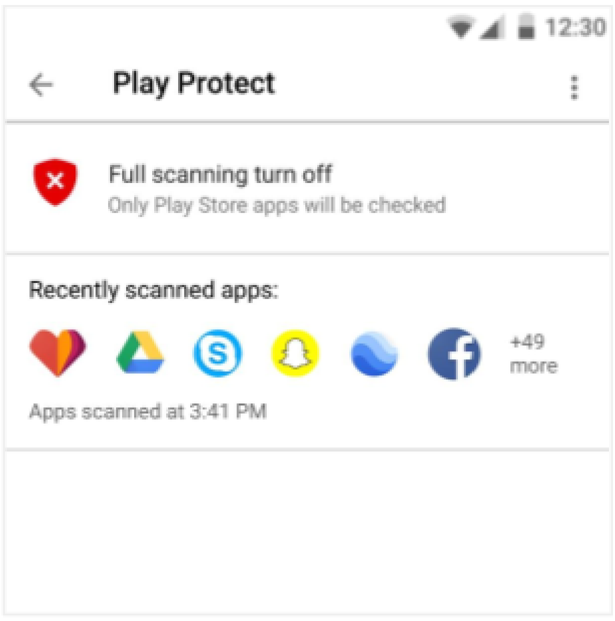
Scan device for security threats

Google will regularly check your device and prevent or warn about potential harm

Improve harmful app detection

Send unknown apps to Google for better detection

Play Protect regularly checks your apps and device for harmful behavior. You'll be notified of any security issues. Learn more


12:30

Play Protect

Full scanning turn off

Only Play Store apps will be checked

Recently scanned apps:

+49 more

Apps scanned at 3:41 PM

Aims & Objectives

- Help figure out the interaction the team needed to consider for the new Team's project.
- Help to clearly visualize main user journeys by creating a prototype using Framer.
- Collaborate in working out missing links/state/user journeys: help identify missing user flows in the project.

Stakeholders

Play Protect team is composed of:

- 3 senior Software engineers
- 1 software engineer
- 2 UXE engineers (including myself)
- 2 UX designers
- 1 intern
- 1 QA & testing specialist

In particular, this project's main stakeholders were the UX team from Play Protect

Timescales & Budget constraints

Timescale	**Budget**
No hard deadline. Completed September 2019	N/A

This project had a timeframe of 3 weeks, and the project was finished in less time than expected.

Methodology

Design

- Understand the flow of the app I needed to recreate on Framer
- Use the Jpeg resources given by UX team to create the prototype

- Recreate all user flows
- Recreate all entry points:
  - Notification entry point
  - Settings
  - Play Store
  - My Apps & Games

Implementation

I had to learn how to use Framer and CoffeeScript to begin with. After I got the basic concepts I started playing around with the tool.

- Upload all image files to Framer
- Have scrollable views
- Create a fixed bars:
  - Status bar
  - Navigation bar
- Create overflow menu with the proper animations to show and hide
- Create setting functionalities for the overflow menu
  - Turn off Play Protect
  - Confirmation dialog
- Create all functionalities needed
  - When Play Protect is turned off from settings, display a red icon card on Play Protect home. And when turned on, display a green icon card.
  - When a card changes, the layout should adapt to the space of the new card

Testing & Verification

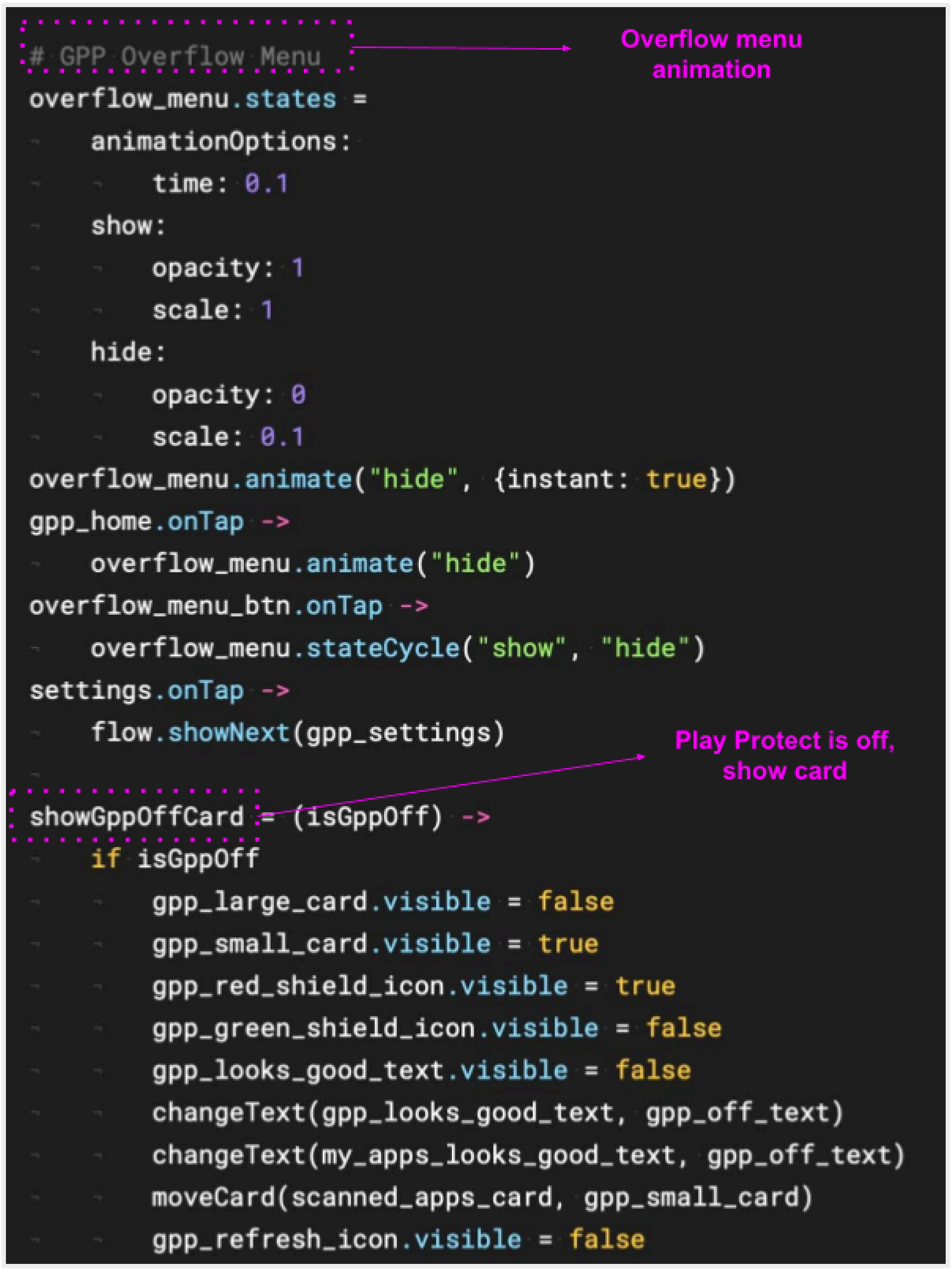
For this project there wasn't a formal "testing plan" put in place. In a way, this project was the actual testing of the user flow and entry points of another project (project inception!) The testing for this prototype was done manually, by using the prototype and verifying all functionalities needed were properly created. The UX designers tested the prototype daily throughout the creation process. I used a tool called "Transporter": It allows users to quickly and easily distribute mobile web based prototypes to any Googler, anywhere over the air. Transporter ensures that the prototypes being served are always available, and always current.

Tools

The software tools used in this project:

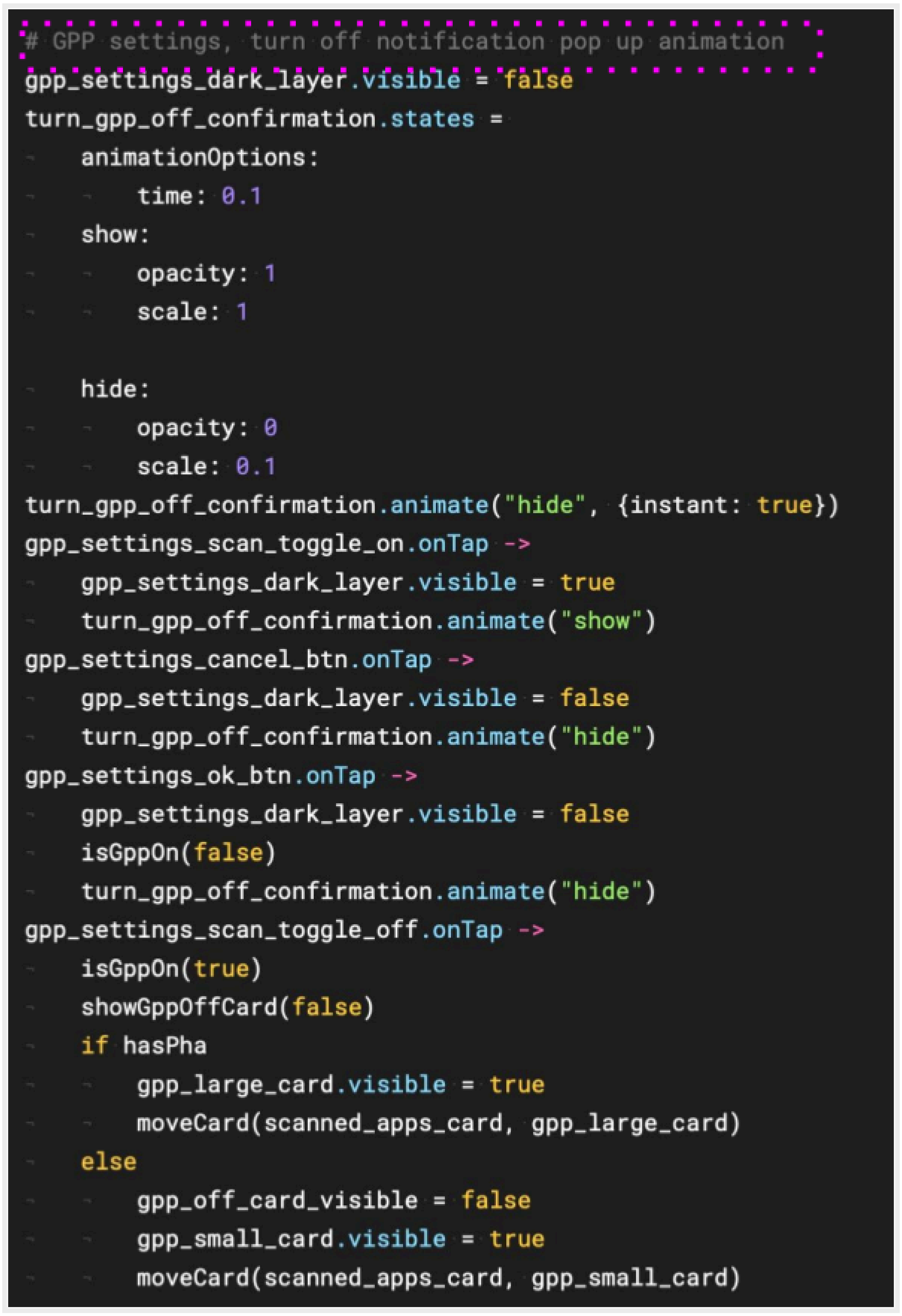
- Framer prototyping tool
- Coffee script language
- New design specifications
- New design assets for prototype
- Transporter tool for uploading prototypes

Evidence



```
# GPP Overflow Menu
overflow_menu.states =
    animationOptions:
        time: 0.1
    show:
        opacity: 1
        scale: 1
    hide:
        opacity: 0
        scale: 0.1
overflow_menu.animate("hide", {instant: true})
gpp_home.onTap ->
    overflow_menu.animate("hide")
overflow_menu_btn.onTap ->
    overflow_menu.stateCycle("show", "hide")
settings.onTap ->
    flow.showNext(gpp_settings)

showGppOffCard = (isGppOff) ->
    if isGppOff
        gpp_large_card.visible = false
        gpp_small_card.visible = true
        gpp_red_shield_icon.visible = true
        gpp_green_shield_icon.visible = false
        gpp_looks_good_text.visible = false
        changeText(gpp_looks_good_text, gpp_off_text)
        changeText(my_apps_looks_good_text, gpp_off_text)
        moveCard(scanned_apps_card, gpp_small_card)
        gpp_refresh_icon.visible = false
```



```
# GPP settings, turn off notification pop up animation
gpp_settings_dark_layer.visible = false
turn_gpp_off_confirmation.states =
    animationOptions:
        time: 0.1
    show:
        opacity: 1
        scale: 1

    hide:
        opacity: 0
        scale: 0.1
turn_gpp_off_confirmation.animate("hide", {instant: true})
gpp_settings_scan_toggle_on.onTap ->
    gpp_settings_dark_layer.visible = true
    turn_gpp_off_confirmation.animate("show")
gpp_settings_cancel_btn.onTap ->
    gpp_settings_dark_layer.visible = false
    turn_gpp_off_confirmation.animate("hide")
gpp_settings_ok_btn.onTap ->
    gpp_settings_dark_layer.visible = false
    isGppOn(false)
    turn_gpp_off_confirmation.animate("hide")
gpp_settings_scan_toggle_off.onTap ->
    isGppOn(true)
    showGppOffCard(false)
    if hasPha
        gpp_large_card.visible = true
        moveCard(scanned_apps_card, gpp_large_card)
    else
        gpp_off_card_visible = false
        gpp_small_card.visible = true
        moveCard(scanned_apps_card, gpp_small_card)
```

Skills	Evidence
Skills: 1. Create effective and secure software solutions using contemporary software development languages to deliver the full range of functional and non-functional requirements using relevant development methodologies. 2. Produce high quality code with sound syntax in at least one language following best practices and standards. 3. Deliver software solutions using industry standard build processes, and tools for configuration management, version control and software build, release and deployment into enterprise environments. **Technical knowledge:** 1. How teams work effectively to develop software solutions embracing agile and other development approaches. 2. How to interpret and implement a design, compliant with functional, non-functional and security requirements. 3. How to use and apply the range of software tools used in Software engineering. **Behavioural skills:**	**Feedback:** **Raissa D G - Host / Tech lead / co -worker Feedback** Impact: small Veronica joined the UX team for a 3-week rotation as a UX Engineer, where she was tasked with creating a prototype for the entire project. At this current stage, the core project was already well-defined, though we were in the process of ironing out the user flows as eng started building the client changes. While I marked Veronica's impact to this project as small as the prototype itself had little to no contribution to the engineering changes going into the project, I'd like to note that her work helped us identify a missing user flow at this early stage. This helped us correct course quickly and prevented an issue that could have been discovered much later, saving us time from having to fix bugs by preventing them in the first place. I was impressed with Veronica's willingness to take on a task with so much unknown to her, such as this prototyping assignment. She quickly ramped up on the project; learning as much as she could about the project's background, the designs and user flows she had to work with, learning how to use prototyping tools after being given an old protype and the tool's website. She was able to deliver the prototype on time, and was able to demo her progress to our team on a daily basis.

1. Able to give and receive feedback constructively and incorporate it into his/her own development and life-long learning. 2. Able to put forward, demonstrate value and gain commitment to a moderately complex technology-oriented solution, demonstrating understanding of business needs, using open questions and summarising skills and basic negotiating skills.	

Conclusion

Outcomes

Have the goals been achieved? Why?
Even though this project was very small and only took me a short time to complete, I can say that it was a success and it had a larger reach than I initially thought it would have.
After finishing my prototype, UX designers realised, thanks to my work done, that they had missed an important user flow. Because this was discovered at an early stage, they managed to do the changes to fix this, avoiding a lot of going back and forward with the engineering team.

Insight Opinion

What did you learn from key learning experiences?
The most important lesson I learnt was that it doesn't really matter if I'm asked to contribute in a very small way to a project, it can still have a great impact on the work of the team. This was a very important lesson for me.
How have you grown because of these experiences?
This project helped me grow as a UX developer, widening my horizons, as I learnt to do something I've never done before (code in a different language I'm not used to). It also

helped my self esteem to grow and feel more confident, since the work I did had a bigger impact than expected, and this made me feel like an important part of the team.

Is there anything you might do differently next time?

I think I did everything pretty well on this project. I delivered my work within the timeframe given, which is good, and I handed in a very detailed and complete prototype. So I wouldn't do anything different next time.

Analysis

What were key factors that shaped this experience?

The key factor that shaped my experience was having to work on a tool I've never worked before, in a language I never used before.

How has this experience changed you as an individual?

It made me realise that the size of the project I'm assigned doesn't really matter. In this case the project was very small with small impact, and because of my thorough work, it ended up being much more important than expected, with a bigger impact than expected for the UX design team.

What factors could have changed the outcome of this experience?

If my work didn't "discover" the missing user flows, perhaps it wouldn't have made such a big impact for me, and of course for the team. The fact that I was recognised for this work, and was congratulated by my team was far more important for me than the work I did on its own. It was a lovely experience overall.

Re-enable suspended apps project

Task title	**Re-enable suspended apps on Play Protect**
Apprentice role	UX Engineer
Company's name	Google
Date of submission	11th of August, 2019

Introduction & Background

Play Protect Home already warns the user about Potentially Harmful Apps (malware) installed on their device, but no warnings are shown about other categories of problematic apps:

- Apps blocked / suspended in Play

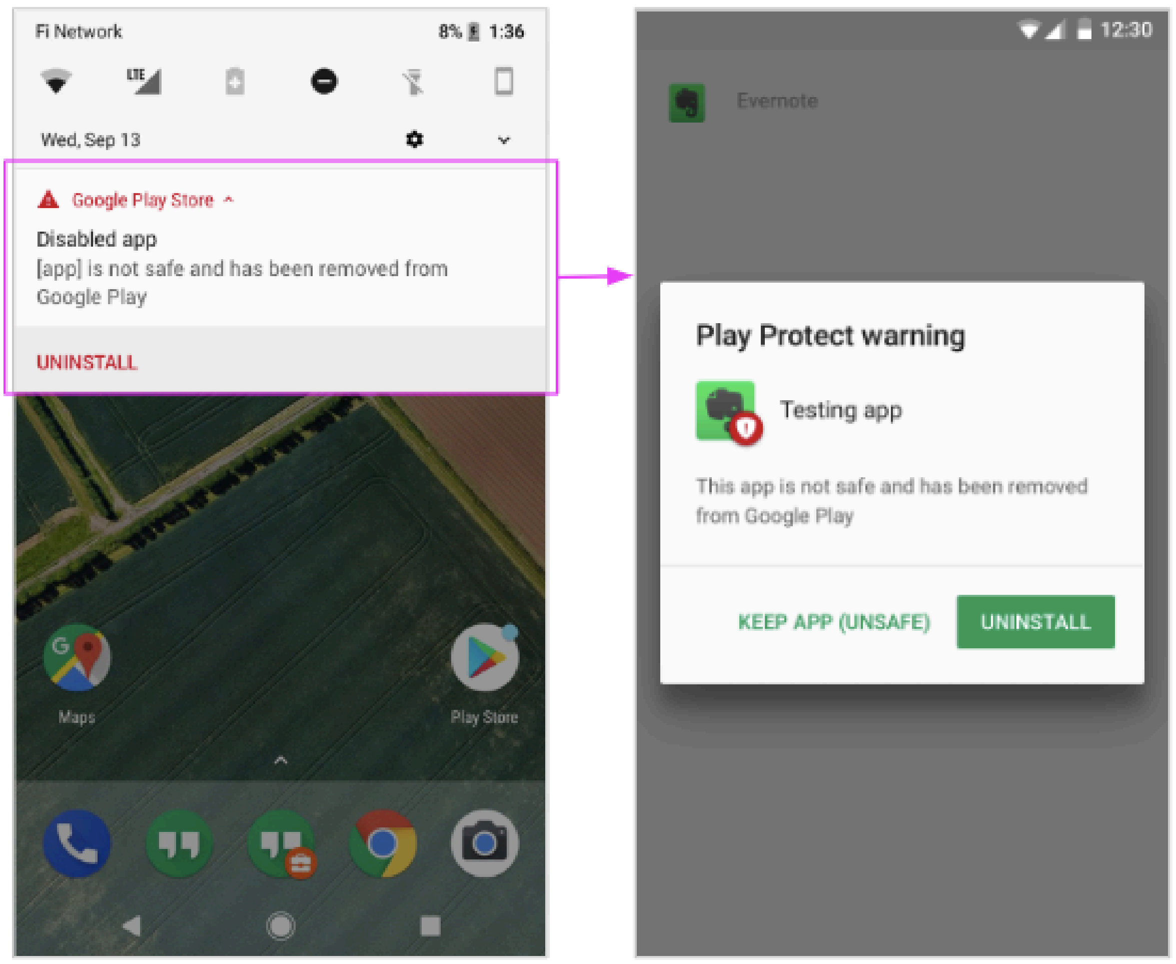
We believe that these apps are bad for user experience in varying degrees, and want to display and execute appropriate actions on users' devices in Play Protect Home and other surfaces to inform the user and help them clean up their device (e.g. uninstall these apps). We currently disable and suggest to the user that they uninstall the unsafe apps we've already suspended from Play. We now want to give the user the option to be able to keep these apps if they want to.

Suspended or blocked apps can be:

- Offensive (e.g. Kids apps showing porn ads)
- Deceptive (e.g. Bank or retail impersonator)
- Degrading Android UX (e.g. Popping up ads in the background)

What is a blocked or suspended app

- For Play Store users
    - Suspended and blocked apps are the same.
    - Apps can't be searched and downloaded.
    - In-app purchases and subscriptions are disabled.
- For developers
    - Suspended apps can't be edited. Appeal is the only way to reinstate it.
    - Blocked apps can be re-submitted for manual review (self-remedy)


Fi Network 8% 1:36

Wed, Sep 13

Google Play Store

Disabled app

[app] is not safe and has been removed from Google Play

UNINSTALL

Play Protect warning

Testing app

This app is not safe and has been removed from Google Play

KEEP APP (UNSAFE) UNINSTALL

Aims & Objectives

Feature Description

Play Protect home should support showing suspended or blocked apps by adding new Uninstall recommendations. Through these cards, users will be able to:

- Uninstall or re-enable suspended (auto-disabled) apps.

The data that would be used by Play Protect Home to generate these warnings already exists in the code base. The new UI that would be added in Play Protect home, as well as updates to existing UI to support the required user flows are as follows:

- To warn users if they have suspended/blocked Play apps on their device. We believe these are bad for user experience.
- To allow the user to easily uninstall these apps, or keep them under their own risk
- To have Play Protect **not** warn again if an disabled app was re-enabled. Change status of the app from disabled to "re-enabled".

Stakeholders

Play Protect team is composed of:

- 3 senior Software engineers
- 1 software engineer
- 2 UXE engineers (including myself)
- 2 UX designers
- 1 intern
- 1 QA & testing specialist

Timescales & Budget constraints

Timescale	**Budget**
No hard deadline. Completed August 2019	N/A

There was no hard deadline for this project, although it was expected to be finished in a few weeks time.

Methodology

Design

The project will be split into the following stages for easier implementation:

When the user opens Play Protect Home:

- Play Protec fetches a security report from the Security Status API. The security report contains all bad apps identified by the verifier.
- Play Protect renders a new card cluster containing suspended/blocked Play apps. It shows a card for each such app, allowing the user to uninstall or keep the app. The rendering is done by a new controller.
- Uninstalling the app calls the existing API for uninstalling, which is already used for malware uninstalls.
- Keeping the app calls a new API, which stores the user's decision in the database.
- The re-enabled app changes status so Play Protect doesn't warn about this app again in the future.

- The app warning will change from RED (disabled app) to YELLOW (re-enabled).

UI Changes

Main user flows:

- **Uninstall app:** Warning card shown about app => user clicks uninstall button => we uninstall app
- **Keep app:** Warning card shown about app => user clicks “keep app” button => we ask user if they’re sure => if yes, we store user’s decision in the verifier’s DB
- After the app is kept (re-enabled) the status of the card changes from **red** to **yellow**.

Implementation

- Create a method to not warn again after the app has been re-enabled. If the app doesn’t warn again, even if it continues to be an Unsafe app, the “unsafe apps warning filter” doesn’t pick the app up as an “unsafe” app, as we want to. Instead it ignores it. This way the warning shown for this card won’t be red anymore.
- “Unsafe apps warning filter” class checks if the method “don’tWarnAgain” has been set, and if true, then it assumes the user doesn’t want to be warned again (since they enabled the app on purpose).
- What I need to do is differentiate that if the app is an enabled one, then it should continue warning but not as a disabled but as an enabled app.

Testing & Verification

Functional tests

1. initialize_twoSuspendedApps
    a. Install one app
    b. Install another app
    c. Create cards
    d. Check that when experiment ON the cards are disabled and shows disabled card / if not, shows “looks good”
2. enableOneSuspendedApp_displaysRedWarningCard
    a. Experiment ON
    b. Install one suspended app

c. Create cards
d. Check right disabled card is displayed
e. Click enable button
f. Accept confirmation dialog and enable app
g. Verify red warning card displays.

Tools

The software tools used in this project:

- Android Studio IDE
- Java coding language
- Internal CI (continuous integration)
- Terminal

Evidence

Skills	Evidence
Skills: 4. Create effective and secure software solutions using contemporary software development languages to deliver the full range of functional and non-functional requirements using relevant development methodologies. 5. Produce high quality code with sound syntax in at least one language following best practices and standards. 6. Deliver software solutions using industry standard build processes, and tools for configuration management, version control and software build, release and	**Niko B - Peer Feedback** **Impact: Critical** Verónica mostly worked on Java, so my feedback here will be focused on Java. As described below, her smaller and more focused changes adhered to our internal Java style guide and best practices. I think this is evidenced fairly well in unit tests: changes are focused, new tests are added when necessary, and they are clean and are adequately named and structured. Code quality does take a hit once changes get larger and less focused. This can be evidenced by how much feedback these

deployment into enterprise environments. 7. Test code to ensure that the functional and non-functional requirements have been met. **Technical knowledge:** 4. How teams work effectively to develop software solutions embracing agile and other development approaches. 5. How to interpret and implement a design, compliant with functional, non-functional and security requirements. 6. How to use and apply the range of software tools used in Software engineering. 7. How to perform functional and unit testing. **Behavioural skills:** 3. Able to give and receive feedback constructively and incorporate it into his/her own development and life-long learning. 4. Able to put forward, demonstrate value and gain commitment to a moderately complex technology-oriented solution, demonstrating understanding of business needs, using open questions and summarising skills and basic negotiating skills.	changes receive during code review as opposed to smaller ones. To be clear, more senior engineers will also struggle to maintain and submit larger changes, but they also tend to avoid authoring large changes altogether, opting instead to break the problem down into smaller chunks, and submitting multiple bite-sized changes rather than a single XL one.

Conclusion

Outcomes

Have the goals been achieved? Why?

The goals for this project were successfully achieved. Users are now able to re-enable apps that were automatically disabled by Play Protect. These apps, once kept on the device, won't be showing a red warning prompting the user to uninstall such apps.

Insight Opinion

What did you learn from key learning experiences?

Understanding the whole Play Protect security system was a huge challenge for me. Understanding what all the classes did, and what was the flow to reinstall and keep apps was very complex. After going through the code many times, I finally understood how everything worked. I had to ask my peers multiple times to clarify things.

How have you grown because of these experiences?

I've definitely grown as a developer after working on this project. The challenges are increasing exponentially as I move from one project to the next: tasks become harder, and I'm expected to work more independently each time.

Is there anything you might do differently next time?

I think this project went swiftly. I applied all peer feedback regarding splitting the tasks into smaller pieces so there's never one big list of changes to submit, rather smaller changes that can be easily approved by peers.

Analysis

What were key factors that shaped this experience?

Making changes to the UI was already something I did many times in previous projects, so this wasn't a big challenge for me. Of course, there's always something new to learn about UI, but it wasn't significant. What was a huge challenge was to understand how the security logic worked on Play Protect. How apps are verified, how the warning levels are identified, etc. Understanding all the logic behind Play Protect was the main thing.

How has this experience changed you as an individual?

Now that I know there's so much work behind Play Protect, I certainly have more respect for the app. I've learnt key things during this project like working with peers when things become too complex. Recognising that my software engineer skills were not enough at this point to work completely independently.

What factors could have changed the outcome of this experience?
If I were the only person working on this project I wouldn't have been able to complete it. Definitely having other more senior software engineers working side by side with me made all the difference. This project was too complex for me to have been able to handle it myself.

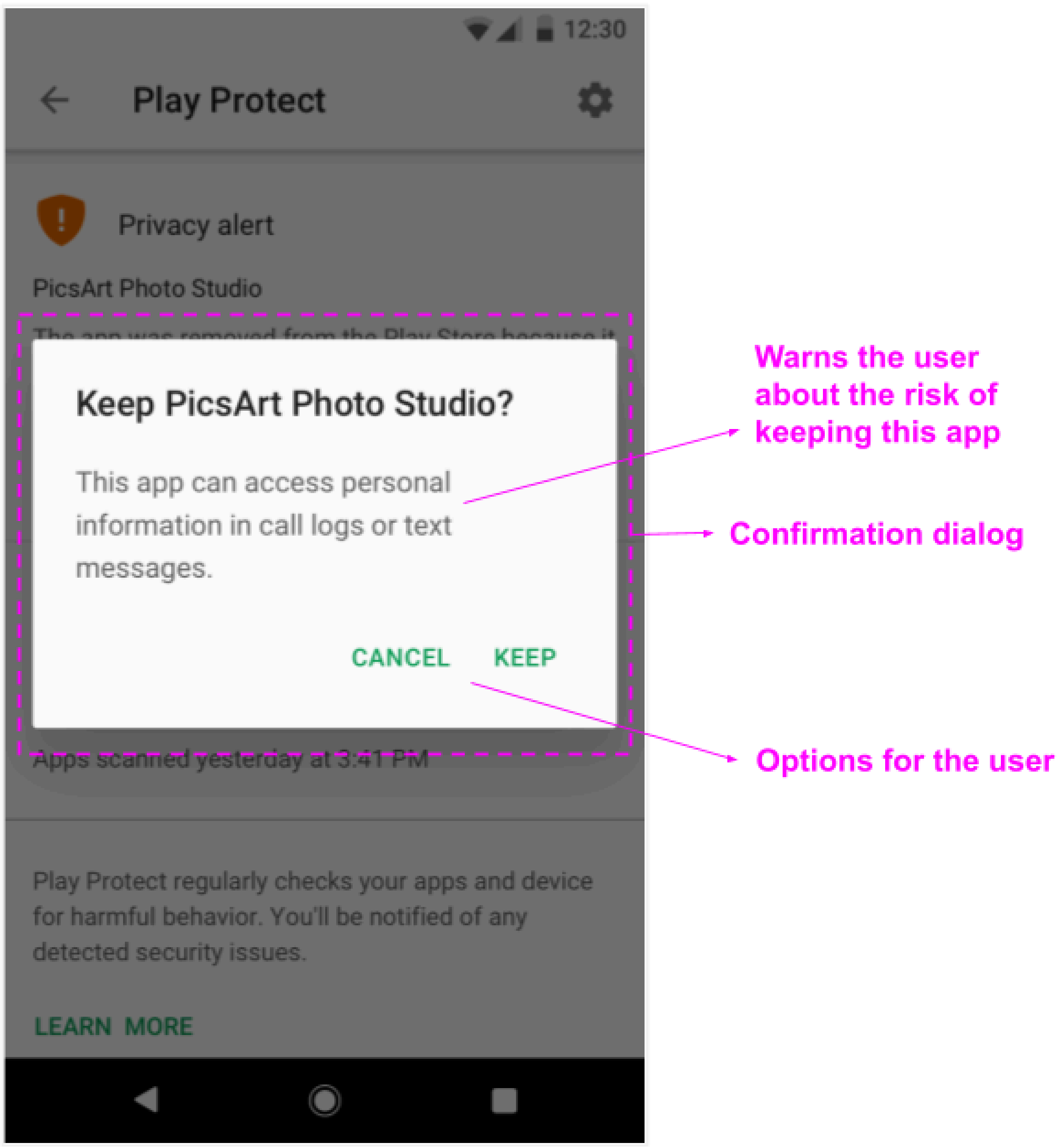
Alert dialog box to confirm user action

Task title	**Alert dialog box to confirm user action**
Apprentice role	Software engineer | UXE engineer
Company's name	Google
Date of submission	17th of December, 2019

Introduction & Background

This project is an extension from the previous project to "**re-enable suspended apps on Play Protect**". After user research and experimentation, we realised the user needed extra confirmation when re-enabling the suspended apps on their device.
The solution we found for this UX issue is that we want to make it perfectly clear to the user about the risks of keeping suspended or blocked apps, by means of a confirmation dialog.

This confirmation dialog appears after the user clicks on “keep app” on the warning card


12:30

Play Protect

Privacy alert

PicsArt Photo Studio

The app was removed from the Play Store because it

Keep PicsArt Photo Studio?

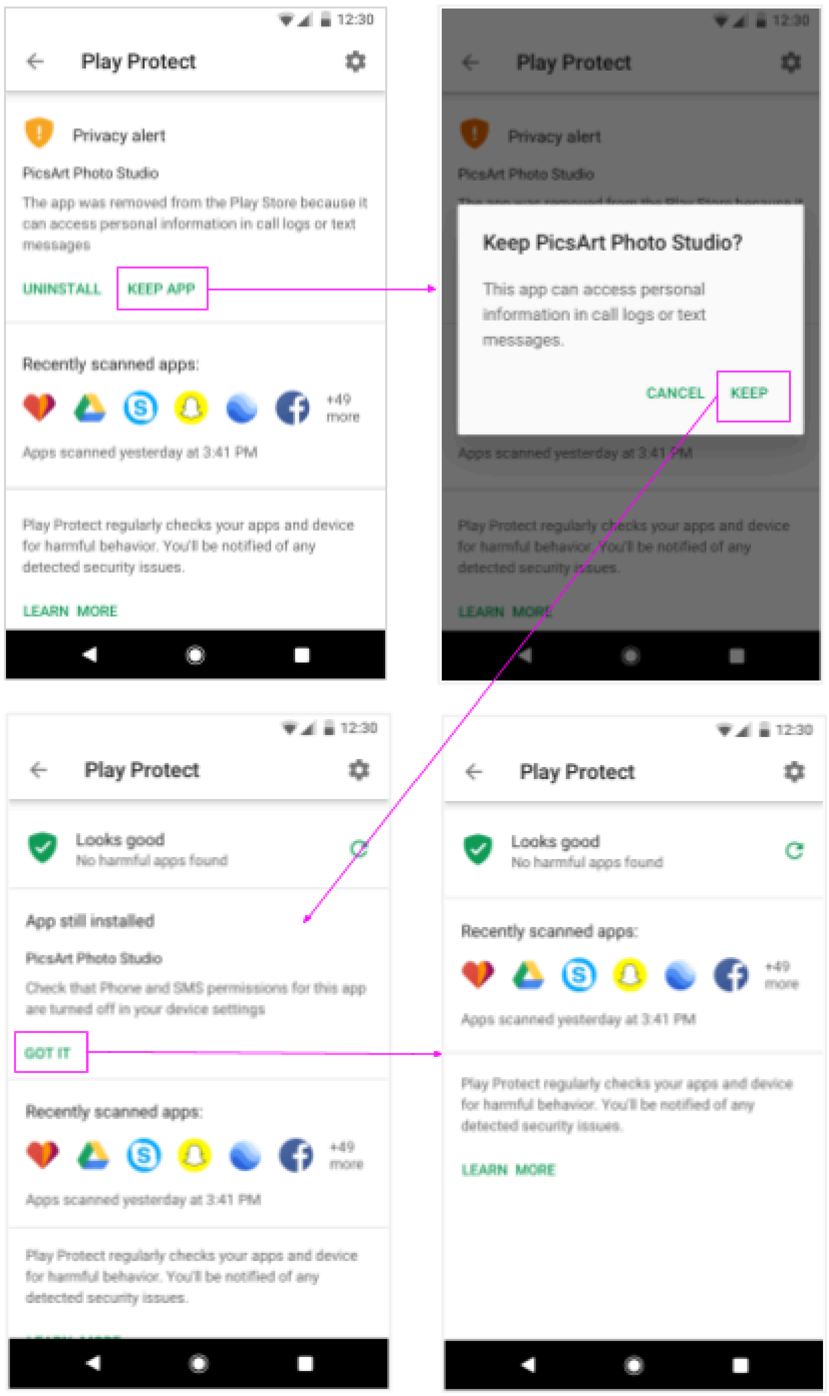
This app can access personal information in call logs or text messages.

Apps scanned yesterday at 3:41 PM

Play Protect regularly checks your apps and device for harmful behavior. You'll be notified of any detected security issues.

Aims & Objectives

- Adding new warning cards in Play Protect Home
- A “keep app” flow, encouraging users to uninstall the app
- A confirmation card to acknowledge user action


12:30

Play Protect

Privacy alert

PicsArt Photo Studio

The app was removed from the Play Store because it can access personal information in call logs or text messages

UNINSTALL KEEP APP

Recently scanned apps:

+49 more

Apps scanned yesterday at 3:41 PM

Play Protect regularly checks your apps and device for harmful behavior. You'll be notified of any detected security issues.

LEARN MORE

12:30

Play Protect

Privacy alert

PicsArt Photo Studio

Keep PicsArt Photo Studio?

This app can access personal information in call logs or text messages.

CANCEL KEEP

Apps scanned yesterday at 3:41 PM

Play Protect regularly checks your apps and device for harmful behavior. You'll be notified of any detected security issues.

LEARN MORE

12:30

Play Protect

Looks good
No harmful apps found

App still installed

PicsArt Photo Studio

Check that Phone and SMS permissions for this app are turned off in your device settings

GOT IT

Recently scanned apps:

+49 more

Apps scanned yesterday at 3:41 PM

Play Protect regularly checks your apps and device for harmful behavior. You'll be notified of any detected security issues.

12:30

Play Protect

Looks good
No harmful apps found

Recently scanned apps:

+49 more

Apps scanned yesterday at 3:41 PM

Play Protect regularly checks your apps and device for harmful behavior. You'll be notified of any detected security issues.

LEARN MORE

Stakeholders

Play Protect team is composed of:

- 3 senior Software engineers
- 1 software engineer
- 2 UXE engineers (including myself)
- 2 UX designers
- 1 intern

- 1 QA & testing specialist

Timescales & Budget constraints

Timescale	Budget
No hard deadline. Completed December 2019	N/A

By this time Covid-19 was already a problem in the UK and as I was pregnant at the moment I had to take some time off work and also do a lot of work from home. This shaped the time the project took. I ended up taking early maternity leave, so I couldn't finish the whole end to end process. Although I covered most of the testing, I couldn't finish with all, neither could monitor metrics.

Methodology

Design

The design of this project will be split into

- Implement the library AlertDialog to create a new dialog to display the confirmation dialog
- Create new strings as needed for the dialog
- Store the user's decision when they choose to "keep" the app.
    - On success: show the confirmation dialog
    - On failure: show a dialog "Could not store 'keep app' user decision"

Part 1

Make all UI related changes

1. Change title of status card to "Unsafe app disabled"
2. Change text fields in confirmation dialog according to detox design:
    a. change title
    b. change body
    c. button text
3. Build 'app kept' card data :

   a. Following design:
      i. Add title
      ii. body
      iii. warning level

Part 2

Enable app workflow

1. Add the logic to re-enable the app
2. Add abstract method to Controller
   a. onAppKept().
3. In the Controller call the override onAppKept() and inside use enablePackage() to re-enable the app.
4. In MediumLevelClusterController call onAppkept()

Implementation

I started by getting familiar with the AlertDialog Android library. This library is very easy to implement, where you have to set the message to display, and the buttons with their corresponding action.

I then had to get familiar with Java Futures and how to use them.

Store the user's decision to keep the suspended app

- Create the new dialog
- On the warning card, when the user selects to "keep" the app, the dialog is displayed with the corroboration of their choice. The user can either continue to keep the app or cancel the action.
- Add the keepApp function that will receive an "unsafe app" and will return a FluentFuture
- If the user decides to keep the app (clicks on "keep app" button) the keepApp function is called and a Java Future (Static utility methods pertaining to the Future interface) is triggered. This registers separate success and failure callbacks to be run when the Future's computation is complete or, if the computation is already complete, immediately.
- After the user's selection is recorded in the controller, then Play Protect shouldn't warn about this same app again in the future, as the user already chose to keep it

Testing & Verification

Tools

The software tools used in this project:

- Android Studio IDE
- Java language
- Android's AlertDialog library
- Internal CI (continuous integration)
- Terminal

Evidence

Skills	Evidence
Skills: 8. Create effective and secure software solutions using contemporary software development languages to deliver the full range of functional and non-functional requirements using relevant development methodologies. 9. Produce high quality code with sound syntax in at least one language following best practices and standards. 10. Deliver software solutions using industry standard build processes, and tools for configuration management, version control and software build, release and deployment into enterprise	**Raissa DG - Host feedback** Veronica's work on the dialog box is part of a crucial flow in one of the possible user journeys in our Security app. As part of this work, she had to consult with our UX designer and the tech lead who had built the event callbacks to ensure that her work was in line with the technical design. Veronica worked on this independently entirely, as our point of collaboration for this project ended with a simple dialog API design. She did very well on this, delivering the complete interface on time. **Niko B - Peer feedback** Verónica added unit and functional (when

environments. 11. Test code to ensure that the functional and non-functional requirements have been met. **Technical knowledge:** 8. How teams work effectively to develop software solutions embracing agile and other development approaches. 9. How to interpret and implement a design, compliant with functional, non-functional and security requirements. 10. How to use and apply the range of software tools used in Software engineering. 11. How to perform functional and unit testing. **Behavioural skills:** 5. Able to give and receive feedback constructively and incorporate it into his/her own development and life-long learning. 6. Able to put forward, demonstrate value and gain commitment to a moderately complex technology-oriented solution, demonstrating understanding of business needs, using open questions and summarising skills and basic negotiating skills.	needed, e.g for Detox) tests to ensure her changes were correct and to avoid regressions. Her tests were often clear, short and precise, and served fairly well as additional documentation of the system's behaviour. Verónica's unit tests usually followed best practices such as: ● Only one behaviour tested per unit test ● Include behaviour, condition, and expectation to the test name ● Keep tests short and focused ● Visually split the test into arrange/act/assert sections. Unfortunately Verónica did not have the opportunity to delve into end-to-end testing, nor monitoring and metrics analysis to ensure non-functional requirements were met due to maternity leave.

Conclusion

Outcomes

Have the goals been achieved? Why?

All goals were achieved: The dialog successfully shows when the user decides to keep an unwanted app (suspended or blocked by Play Protect).

Insight Opinion

What did you learn from key learning experiences?

This project was very useful for understanding Java Futures extensively. Also, this project was the V2 of a previous project. I could see how V1 was done and based my work on that. This helped me maintain the consistency of the code.

How have you grown because of these experiences?

All experiences at Google are growth ones. I'm increasingly working on harder projects, and collaborating everytime a bit more, and needing less guidance each time. I feel it's like going up stairs.

Is there anything you might do differently next time?

I wouldn't change anything on this project. It was a fairly easy one, and I managed to complete it within the expected timeframe. I wasn't able to finish the whole end - to - end process for this project as I had to leave earlier than expected on my maternity leave. This prevented me from finishing the whole testing process and metrics analysis. This work was taken over by other co-workers.

Analysis

What were key factors that shaped this experience?

At the moment I worked on this project, I was already familiar with how my team worked and how they did things. I felt more comfortable at this point, as I already knew my teammates well.

How has this experience changed you as an individual?

As mentioned before, every learning experience or project I work in is a growth experience, not only technical growth, but also personal. Every step in the right direction gives me more confidence in my skills: coding skills, people skills, team work skills, and more.

What factors could have changed the outcome of this experience?

The only factor that could have changed the experience could be a reduced timeframe. I think the available time to finish this piece of work was the right one. If reduced by a week, I don’t think the project would have been as enjoyable.