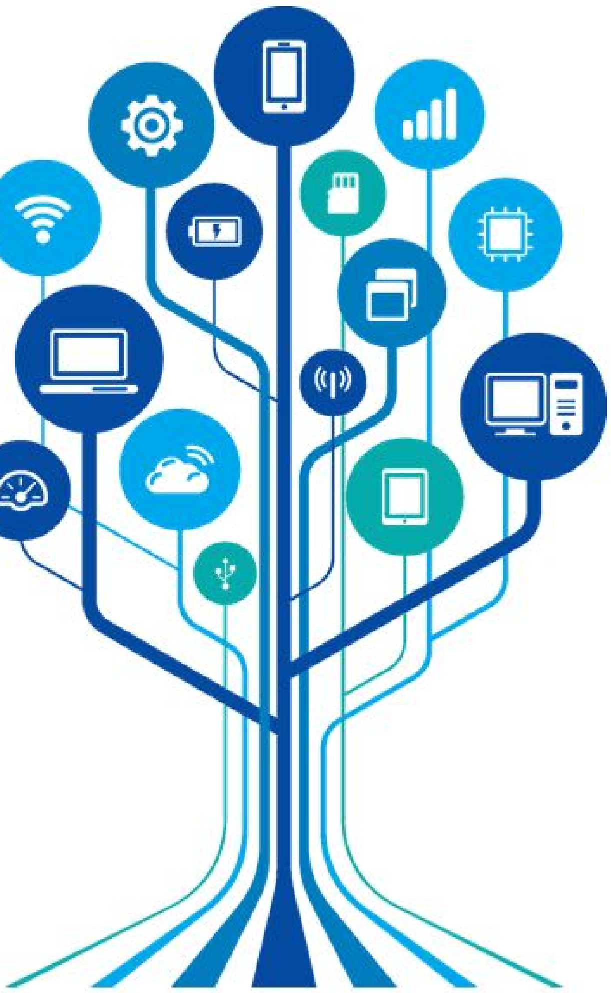
Quick Menu Final Report

Aleksiy Kudelsky, Hayley McLennan,
Ryan Hutton, Travis Wilson, and Arshi Annafi



University of Victoria. March 2016.

Table of Contents

Table of Figures

Problem Description and Motivation

The current app menu system is standard across almost all phones today. It is layed out like an array or bookshelf of applications, usually alphabetically or in the order that they were installed. The user swipes sideways or vertically to switch pages and find the app that they are looking for. It makes sense and it works, so why change it?

The current menu system doesn't allow for the user to optimise their actions. Each action through the menu requires the user to orient themselves. When a user opens the menu it always opens the last page that they were on. This feature is actually a drawback. Users rarely remember what page they were last on, but they have to decide which direction to navigate, relative to this page. A huge array of applications is also overwhelming. The user has to scan through each application to decide if this is the application they are looking for. The steps between deciding on a goal and having that goal fulfilled should be minimized. For example, desktop programs use hotkeys for certain actions. An experienced user learns the hotkeys and doesn't have to navigate through various steps to perform their action. In fact, humans are quite good at minimizing this if the task allows it. For example: an experienced typist does not need to look at the keyboard to type, and they do not need to consciously think about each letter they hit, or its location. They merely think of the word they want, and their muscle memory does the rest.

Currently, the menu system does not allow you to use your muscle memory efficiently. There are some existing solutions, but their application isn't ideal. Hotkeys do not lend themselves to phones. Physical buttons on the phone are limited, and virtual buttons take up valuable screen space. However, one area where phone designers have allowed for muscle memory to be used is the phone lockscreen. Users quickly get used to inputting their pin codes, to the point that they can do it without looking. Elegant alternatives to the lock screen have also become popular, like the Android grid; which is where we take our inspiration. A second problem with the current menu system is organisation. Why force the user to do something that a computer can do better. Applications are already categorised on the app store. They should be organized automatically, and then the user should have the option to customize. These are the the two main problems which QuickMenu aims to solve, along with the goals of being simple and minimalistic in its design and functionality.

Existing Solutions

There are some apps that are similar to our QM, mainly they are found on the Android app store. Apps such as “Folder Organization Lite”, “SwipePad-Gesture Launcher”, and “LMT Launcher (Pie)” all have similar, but unique approaches to the same task.

Folder Organization Lite groups applications together based on their labels (i.e productivity, games, finances, etc) and then puts them into folders. Shown in figure 1.1, Folder Organization Lite has the option of selecting the users preferred apps into each folder. This app solves the issue of organization but does not have the same effect of reducing the total time it takes to open the app. The difference between the Folder Organization Lite and QM is how we access the apps. While the Folder Organization Lite has physical folders on the home screen, QM has layers of folders that is accessed through different pattern combinations off of a 3 by 4 grid.

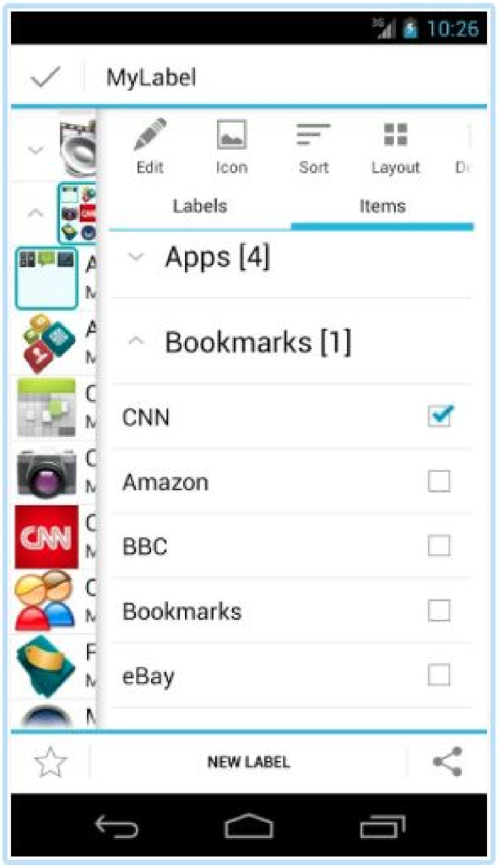


Figure 1.1 Folder Organization Lite

SwipePad is a way for the user to switch apps with a single swipe, the main concept is you can launch anything from within any app. The action of swiping from app to app is similar to QM. It is demonstrated in Figure 1.2 that the user can swipe to a different application in just one gesture. This app has the ability to quickly change from app to app making it very time efficient.

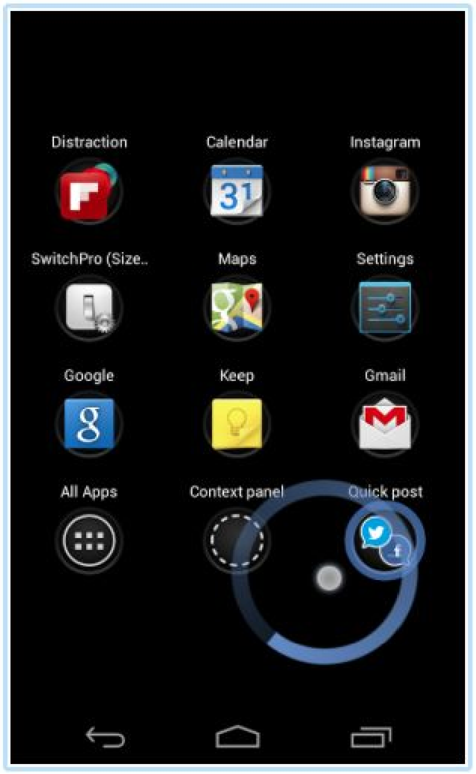


Figure 1.2 SwipePad

LMT Launcher (pie) is a tool used on an android phone that launches a command when the user performs gestures on the screen. To navigate through the LMT Launcher you use a system-wide PieControl and the user can perform a list of predetermined patterns that all have a unique commands. This application allows the user to switch between applications in a quick and easy manner. It lacks the organizational aspect that is included with QM, but it is an easy way for the user to use different patterns to open apps. LMT Launcher (Pie) is the most similar to our QM system, it provides the user with a quick and efficient way to navigate from app to app and it is also gesture-based.

Proposed Solution

Our solution is to have a 3 by 4 grid that categorizes apps into different folders. We will have three initial folders “productivity, communication, and fun” and all apps will be split into these three categories. By categorizing the apps into different folders the user is provided with a way to organize their apps. This solves the issue of having multiple pages of apps because everything is categorized into folders that access by different pattern combinations. Once the user has learnt the different patterns for unlocking apps, it will reduce the amount of time it takes to find an app. The user has the option of using the default patterns or they are able to customize the patterns to whatever they would want. Having patterns for unlocking will allow the user to open their apps in a quick and efficient way.

Personas

To help us decide which direction to go with our app, we developed two personas. These personas were representative of the two main groups of users we are targeting, the tech-savvy power user, and the young, casual technology user. When we had to make decisions for our app, it helped to imagine how this would affect both of these personas. This next section will provide the full backstory to each of our personas.

Robert

Robert is a 20 year old university student, working towards his undergrad. He commutes on the bus, and attends all of his classes, labs, tutorials, and group meetings. Robert's daily used apps are: PDF Viewer, Dropbox, Texting app, Facebook, Facebook messenger, Snapchat, Whatsapp, Calendar, web browser, Todo List, and the music player. He also occasionally uses the mailing, contacts, calling, and note taking apps. He uses an Android™ phone where he can customize the locations of app on the home screen, and also has access to a list sorted from A to Z of all the apps that are installed on his phone. Currently it requires 5 to 7 taps to get to the app he wants, and he is looking for a better solution. Robert wants to reach all ends of his phone fast and easily.

Ashley

Ashley is an 18 year old aspiring artist attending her first year of undergrad. She drives to school everyday in her brand new BMW and loves getting coffee from Starbucks every morning on her way to class. Ashley loves social media, she instagrams a picture of her coffee every morning. Aside from that she loves texting and applications such as Snapchat, Messenger, Facebook, Instagram, Tinder, Pinterest, Spotify and Twitter. She also has Tumblr where she blogs about her daily life. Ashley's iPhone™ is an absolute mess. She downloads new apps daily and never deletes them. She currently has over 100 applications spread over 8 pages. She doesn't use the current organization system because it's too much work.

Scenarios

After creating our personas, the next task was to create scenarios for them. The purpose for these scenarios was to give a general task that our personas could be trying to accomplish. An important thing to consider when creating scenarios is that they should be general descriptions of the task, and as such should not have specific UI details in them.

Scenario One

Richard downloads, the “Quick Menu” app. He has read the description of the app from the app store but is still unfamiliar with how to use it. Before Richard can start using the application he must complete the tutorial on how to use it. The tutorial shows the user how to unlock his Facebook messenger app. It shows the user a hand animation on how to enter the correct passcode. Once Richard is comfortable with the pattern he moves onto the next slide of the tutorial. The tutorial displays a pre-drawn pattern that Richard is prompted to enter. Once he has entered the correct pattern the tutorial is finished

Scenario Two

Rebecca has downloaded the “Quick Menu” app. She has finished the initial tutorial on how to use the app and wants to customize the pattern for unlocking Facebook Messenger. Rebecca goes to her settings and finds the tutorial that explains how to complete her task. The tutorial shows how to select which app the user wants to customize and shows the user how to enter their desired pattern. Once the pattern has been entered, the tutorial is over. After the tutorial, Rebecca is able to go into her settings and successfully customize her pattern for Facebook Messenger.

Scenario Three

Robert is in class. He receives a text from one of his group members. Robert is asked to share a file on Google docs and send the link to everyone on facebook messenger, so that they can edit the document. Furthermore, he is told that there is a group meeting tomorrow. Robert now needs to launch a file management app, upload the file on Google docs, share the link on Facebook messenger, then launch the calendar app and create an event for tomorrow.

Scenario Four

Ashley is on her daily commute, and has just picked up her morning Starbucks. She decides she wants to post an instagram picture of it. Ashley is late to class and irresponsibly tries to take the photo while driving. While she frantically swipes left and right trying to find the correct page and icon among the mess of apps, she drives straight into the fender of an SUV stopped at a red light.

Use Cases

Once some basic prototyping was done, we were able to expand our scenarios into use cases. A use case is similar to a scenario, but has specific user interface details. Since a use case has details about the UI, it is more in depth, and generally longer. The following use cases each correspond to a scenario (scenario one to use case one etc.).

Use Case One

1. The user downloads the “Quick Menu” app from the app store and opens it
2. The system displays a tutorial and the user clicks “Next” to start the tutorial
3. The system shows the default pattern for opening Facebook Messenger
4. The user clicks “Next”
5. The system gives a step-by-step visual animation on how to enter the default pattern to open Facebook Messenger
6. The user clicks “Next” when they are comfortable with the concept
7. The system asks the user “Now you try!” and displays a pre-drawn pattern in red for the user to enter
8. The system makes the user enter in the default pattern
9. Once the pattern is entered the user is able to finish the tutorial by clicking “Done”

Use Case Two

1. The user opens “settings”
2. The user opens “tutorials”
3. The user selects the tutorial entitled “How to customize your patterns”
4. The system displays a tutorial on “Customizing your patterns”
5. The user clicks “Next”
6. The system displays how to customize the pattern for Facebook Messenger
7. The user clicks “Done” once they are comfortable with changing the pattern
8. The tutorial is finished
9. The user opens “settings”
10. The user opens “Customize Patterns”
11. The user selects the app Facebook Messenger to customize the pattern
12. The user enters the desired pattern
13. The user clicks “Done”
14. The new pattern is finalized

Use Case Three

1. QuickMenu is downloaded from the app store, and opened.
2. QM setup process begins. QM scans her phone and uses the App Store’s categories to organize her phone accordingly.
3. Even though QM has been disabled the phone has been tracking which apps Ashley opens the most. QM displays the top 8 and ask Ashley if she feels that these should be mapped to the easiest gestures. Ashley can replace any of them.
4. QM completes the setup process and shows how to the top 8 apps. The rest of the apps can be accessed by navigating through the categories.
5. Ashley now wants to install a new app. She finds it on the App Store and installs it.
6. After the installation process is complete, QM detects a new application has been installed and computes the best place to put it. It shows an animation of the new gesture and asks if Ashley would like to use the calculated location or customize her own.
7. Ashley knows she will be using this application frequently therefore she drags her finger to map her own gesture.
8. QM brings up the gesture screen and allows her to map a gesture. If she replaces another app then QM again calculates the location of the old app and asks her if she wants to customize it.

Use Case Four

1. The user unlocks their phone by entering a premade finger sliding combination on a fourbyfour grid.
2. The user opens the Quick Menu by sliding up from the bottom of the screen.
3. Without lifting his finger he slides toward the social media multicategory
4. He then continues to slide towards the social media category
5. he then slides toward the column containing the facebook icon
6. finally he slides toward the facebook icon itself.
7. The system opens the Facebook application
8. The user is able to select the “more” button option on the Facebook application in the bottom right corner.
9. The user selects “events” and “create”.
10. The system creates the event.
11. The user is able to invite his friends by selecting the “invite” button and is able to search through his friends to see who will be invited.
12. The system will send out invitations to the selected people once the user hits the “invite” button after the one is selected.

Evolution of the prototype

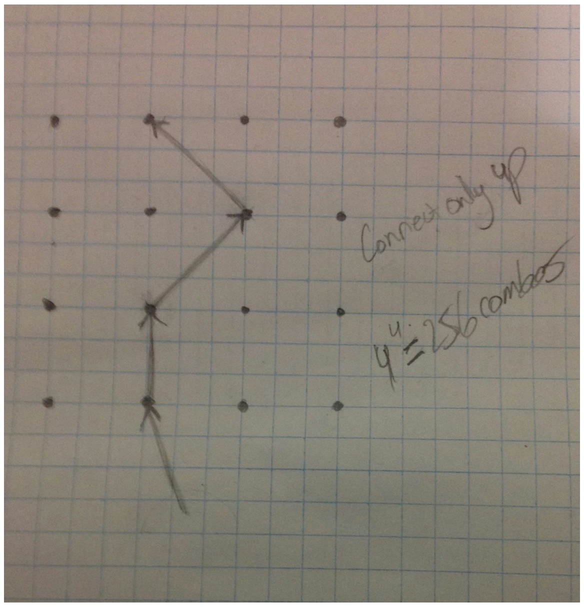


Figure 2.1 - First concept sketch

The general concept was first presented in the sketch above. The grid design borrows heavily from the Android lockscreen. A key difference to is that connections are restricted to one connection on each level. Also, a 4x4 grid was chosen as it allows 256 mappings to be made.

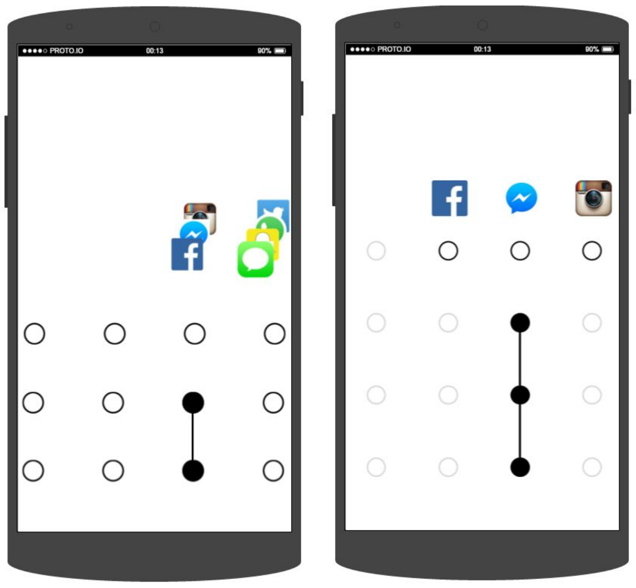


Figure 2.2 - Low Fidelity Prototype Screens

Next a medium fidelity prototype was made to explore some other areas of the design. It is unreasonable to expect a user to memorise all 256 combinations. Therefore, the design also had to allow a user to navigate it and see what applications are available. Here it was decided that the 3rd tier would display four stacked application icons. The final tier would display the application that will be opened.

This was the prototype which was evaluated by users. All users were able to complete the task of opening a specific application in a short amount of time. Feedback given was that a 4x4 grid may be too complex to use on a phone and a tutorial to familiarize a new user.

High Fidelity Prototype

After our low fidelity prototype, a high fidelity prototype was started on Proto.io . Unfortunately for us, Proto.io only had a 15 day free trial period. After the expiration of the free trial period, we moved to the Marvelapp platform to finish the implementation of our high fidelity prototype. Here is a link to our high fidelity prototype: https://marvelapp.com/b0c728 .

The first screen that appears when the prototype is launched is a developer's page as seen in figure 3.1. This screen will not appear in the final implementation. The purpose of this screen is to jump into different sections of the prototype for testing convenience.

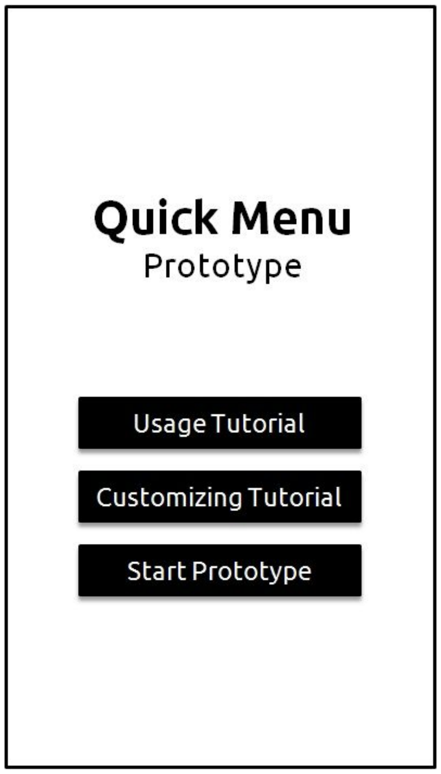
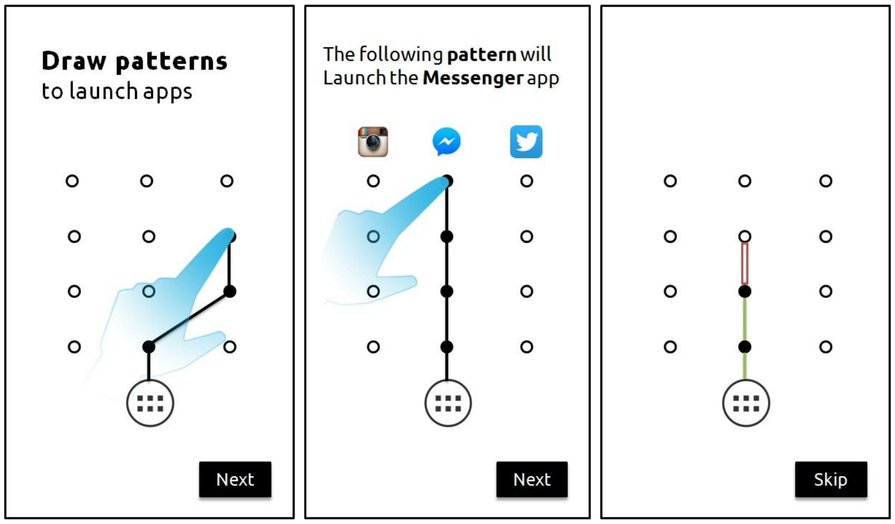


Figure 3.1 - Developers' screen for navigation convenience

In the high fidelity prototype it was decided that a 3x4 grid would allow sufficient mappings (81 distinct patterns) while making navigation much simpler. Tutorials were also designed to familiarize new users with our app and to show them how to use the software. We found it was important for us to provide users with these two following sets of tutorials: how to use the Quick Menu, and how to customize the default patterns.

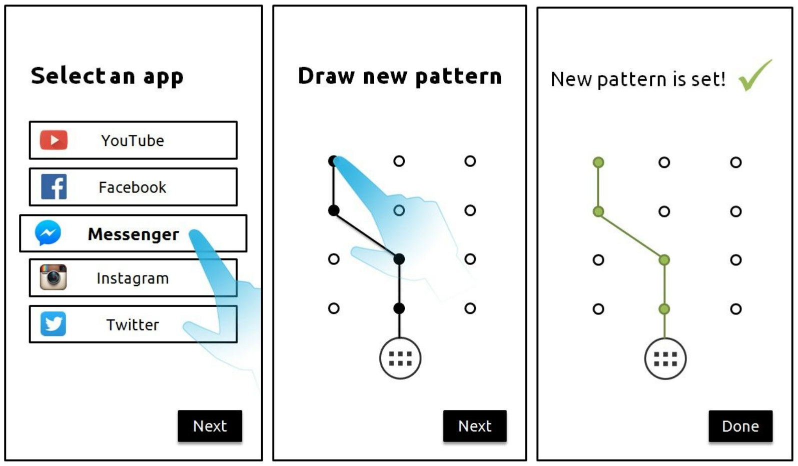
When a user opens Quick Menu for the first time, they will see the usage tutorial. This tutorial introduces the underlying concept of drawing patterns to launch apps as seen in figure 3.2 (a). The tutorial then shows the pattern of Facebook Messenger (figure 3.2-b). After this, The user can either follow the guide to launch Messenger or skip the tutorial and start using the software (figure 3.2-c).



(a) (b) (c)

Figure 3.2 - Screens from the usage tutorial

The other tutorial we have included in the prototype is the customizing tutorial. Users will see this tutorial (figure 3.3) when they would like to customize a pattern. The procedure is simply: select an app, draw a new pattern for the app, and finally confirming the new pattern by tapping on "Done".



(a) (b) (c)

Figure 3.3 - Screens from the Customizing tutorial

Future Work

As we wrapped up our project, we have received positive feedback from our peers and instructors. Some questions raised during our presentation, and suggestions made near the end of the project, lead to a few areas we wish to improve.

One question raised was “what happens if the user wishes to customizes the pattern of an app and the desired pattern is occupied by another app”. Our solution to this problem is that the user will be notified that the desired target pattern is occupied and given the option to either cancel the customization, assign a different pattern for the app in conflict, or overwrite the pattern i.e. leaving the app in conflict patternless. See figure 3.4.

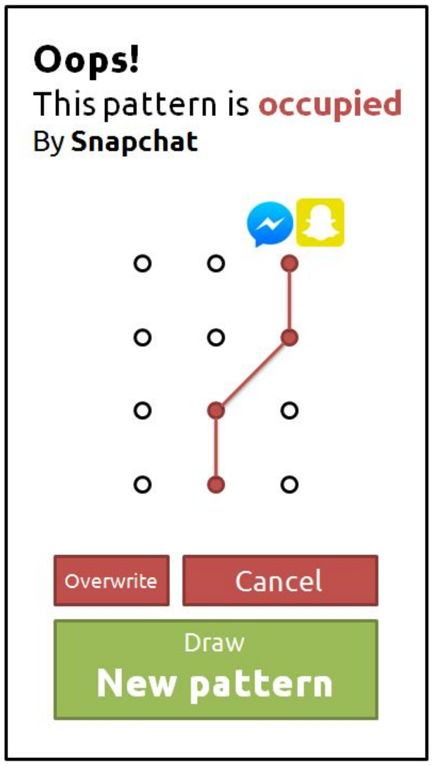
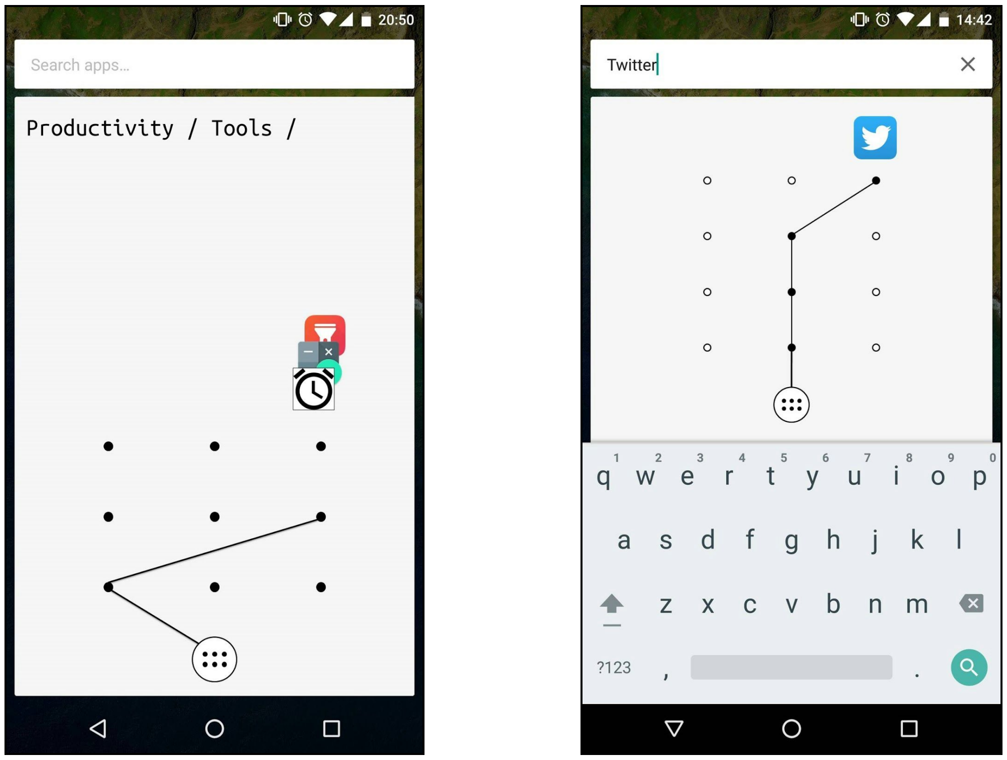


Figure 3.4 - Pattern-occupied-error screen.

Another comment that we received was that once the user is in a level or two deep, there was no indication of the current category. For our solution, we have decided to include a string of text at the top left corner indicating the path of the user's navigation. Figure 3.5 (a) is a screenshot of what this text would look like in deeper levels

Another feature that we believe would be useful to include is a search feature. This would simply display the pattern of the app on the screen as the user searches for a specific app. Figure 3.5 (b) shows this feature in application as the user searches for Twitter.



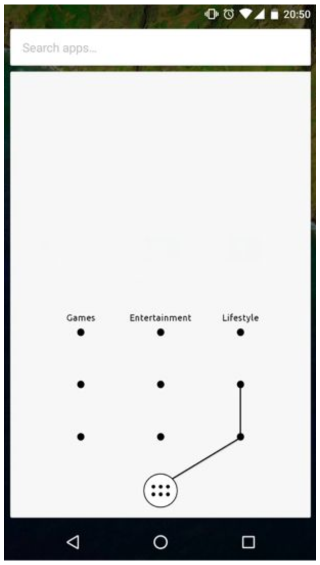
(a) (b)

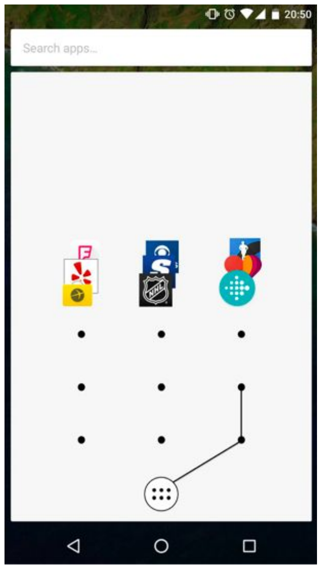
Figure 3.5 - (a) Displaying the path of categories, and (b) Pattern search screen.

Lessons Learned

What may seem simple to a developer, can be beyond complicated for the user. In the early design days of QM we thought it was a simple to learn system, but after user testing, we quickly realized that our concept was not as simple to others as it was to us. This forced us to come up with new ideas, which simplified our system.

In our early prototypes we had a 4x4 grid instead of the 3x4 grid we are currently using. We made this change to reduce the amount of patterns, and to have a horizontally central line. This central line was used to give a simple pattern to frequently accessed apps. Another feedback-induced change from our early prototypes is that we originally had sub categories shown at the third level. With feedback from some user testing we realized that it would help users identify app locations more easily if we showed groups of app icons instead.





(a) (b)

Figure 4.1 - Before (a) and after (b) user feedback.

One thing we learned very early on, is that menu systems are complicated. There's a reason we haven't really migrated away from the standard pages of applications and that's because users have an extremely difficult time understanding the alternatives. When our system was being tested by a new group, many of them had a hard time grasping the concept. After explaining almost the entire system down to its core they could use it, but that is obviously not ideal. With the feedback we received from these users, we made several core changes to our system, and are extremely happy with our final prototype.

Links

This section include links to our high fidelity prototype and the demonstrational video that was created near the end of the project.

High fidelity prototype
https://marvelapp.com/b0c728
Powered by Marvel app

Demonstrational Video
https://www.youtube.com/watch?v=HpkrJtV9KIo

Cover photo: Wall Mantra. The Technology Tree.
http://d3gyiijzpk1c44.cloudfront.net/upload/product_photos/base/0/10/124/original2.687163.1.jpg