Wildbook Application Development Project

Affan Farid, Grae Abbot, Gina Gerace

UIC Computational Biology Lab

2018

Introduction

Each year, 700 million in the world people visit a zoo, but for the most part, the experience limits itself as look and move on. With the increase of technology along with the growing issues of climate change and endangered species, we decided to help each zoo visitor enhance their individual experience while at the same time, learning about the conservation efforts of each of the species.

The goal of this project is to update and add features to a previously created mobile application which uses the camera of a smartphone to give more information and context about specific animals at the zoo, along with being able to take, store, and share pictures of these animals. This will allow a more social, yet informative zoo experience which will captivate the attention of the public for a more enhanced trip.

Wildbook has created software to identify individual animals using unique features of each animals such as stripes on zebras, or flukes on whales. This app is designed to use that software to allow users to identify the animals they see at the zoo, which will be discussed later.

While there are apps that aim to help users identify animals, or plants, or to enhance a zoo visitors experience, none take advantage of Wildbooks image analysis like in this app. Most apps designed for identification provide example images of a certain species of animal or plant to allow the user to compare with their sightings to identify which animal they are seeing. In addition to relying on the user to have a clear view of the animal for a long period of time, they do not provide individual animal identification like in the Wildbook app. The other apps that some zoos provide also only provide certain features like GPS map, or extra information about certain exhibits, but no animal identification. That's where the Wildbook app comes in to provide an interactive user experience that no other app accomplishes.

Currently, there has been a previous build of the Wildbook app created by Alessandro Oddone, who created almost a prototype to what we plan to accomplish. It set out the general framework and structure for this project; the goal is to expand upon this, by taking inspiration from the previous application and starting fresh, creating a brand new application from scratch to complete the final app from this process.

The Project

The requirements of this app are that the user can identify an animal by taking a picture of it using their default phone camera, accessed through the app, and marking the area where the animal that they want identified is by dragging an annotation box around it. The user also has to have access to a gallery of images of animals that the user has identified, with the corresponding information about that animal. In addition, the app has to be able to be used at any zoo.

This app is designed so that the user only has to be able to take a picture and drag an annotation box around the animal that is desired to be identified in order to use the app. Once the user takes the picture of an animal and created the annotation box, the app will identify the animal using Wildbook image analysis, accessed through the IBEIS Java library designed by Alessandro Oddone, shown in Figure 1.

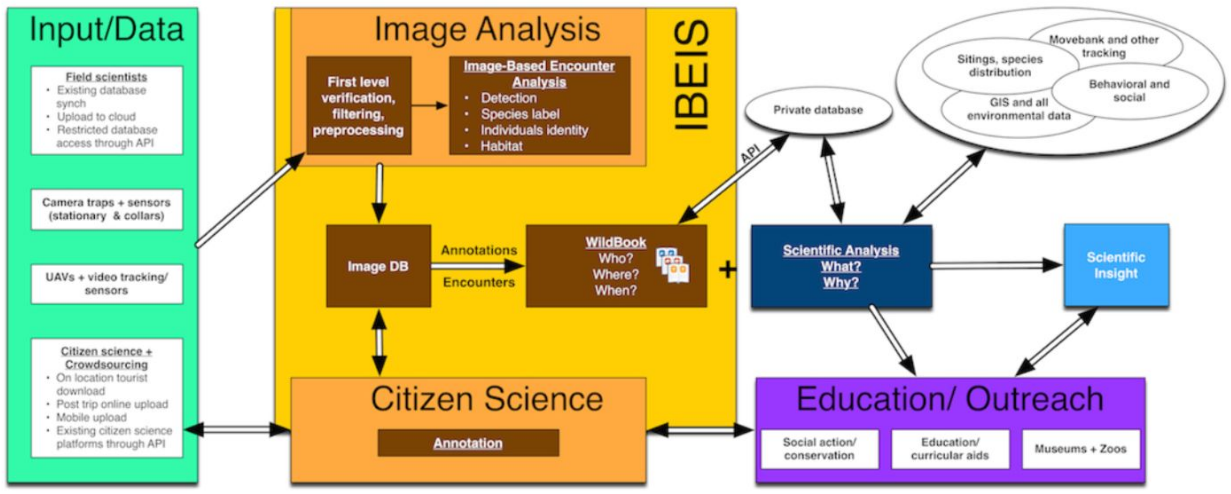
Input/Data
Field scientists
Existing database synch
Upload to cloud
Restricted database access through API
Camera traps + sensors (stationary & collars)
UAVs + video tracking/ sensors
Citizen science + Crowdsourcing
On location tourist download
Post trip online upload
Mobile upload
Existing citizen science platforms through API
Image Analysis
First level verification, filtering, preprocessing
Image-Based Encounter Analysis
Detection
Species label
Individuals identity
Habitat
IBEIS
Image DB
Annotations
Encounters
WildBook
Who?
Where?
When?
+
Citizen Science
Annotation
API
Private database
Movebank and other tracking
Sitings, species distribution
Behavioral and social
GIS and all environmental data
Scientific Analysis
What?
Why?
Scientific Insight
Education/ Outreach
Social action/ conservation
Education/ curricular aids
Museums + Zoos

Figure 1

All of the pictures of the animals that the user has identified can be accessed in the app's gallery. In addition to the image, the gallery will hold information about the animal that can be accessed by touching the thumbnail image in the gallery.

There were multiple decisions that had to be made in the design process process such as: where to put the navigation bar, the format of the gallery, using fragments or activities, the color and user interface design, layout scheme, and functionality.

Design

As much as the team wanted to jump in and start coding all the features, it was quickly discovered that a design process where all the elements and features of the app would be planned out was necessary for both the short term and the long term functionality, including leaving ample room for future use and growth.

First and foremost, the details of what exactly needed to be accomplished were thoroughly discussed in order to base the application design around the use of it; in other words, form would follow function. The team decided on 3 major purposes, to take pictures of animals, to be able to view the pictures taken, and to be able to view detailed information about each animal in the pictures. Using these 3 components, we opted for a simpler, more easy to use application design, as the purposes would then be more concise and more marketable to a wider range of ages, and technological experiences. We wanted to make this application easy to use for anyone and everyone. On top of these three major design features, there was room to grow and add more to the application, as these features would be a relatively stable skeleton to expand if necessary.

At first a basic design layout was created with a top navigation bar of all the different screens of the app. This would make all the features visible at all times, and thus accomplish the simplicity that was needed. An Adobe application, Adobe XD, was used to mock the layout and basic functionality of the application, making it very clear on what we needed to code for eventually. After discussion and research of other popular, well received apps, including the ESPN application, which we took a lot of inspiration from, the team decided on a bottom navbar type layout, as shown in Figure 2.

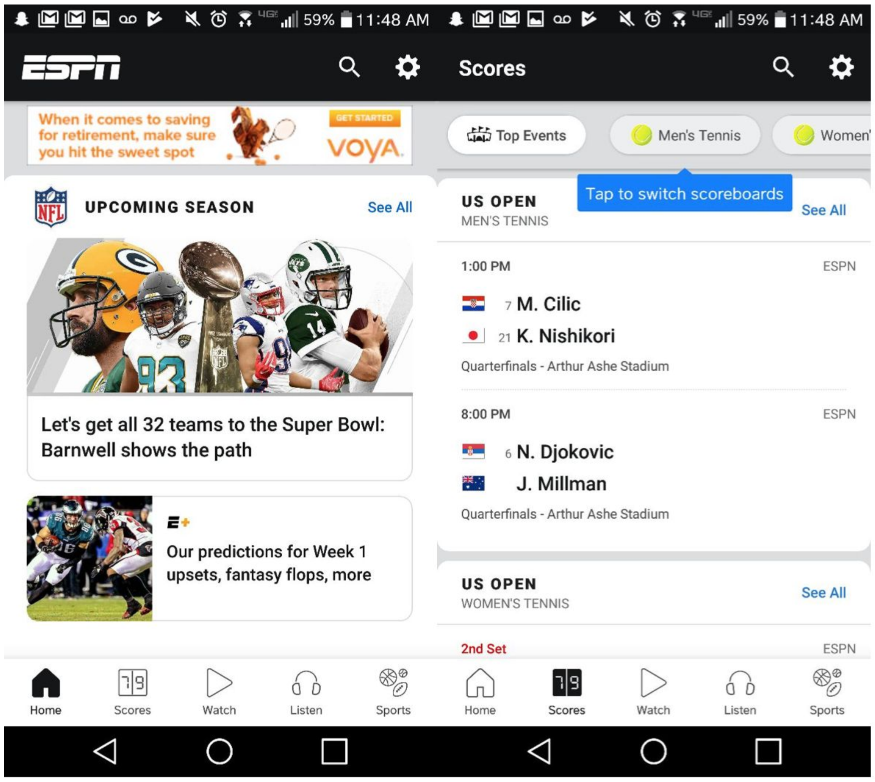

Figure 2

As shown, navbar works in a way where the the navigation bar stays relatively constant, but the actual screen changes based on what item is selected. This allowed for a very easy to use, welcoming overview of the application, yet allowed for detailed to be explored in each of the tabs of the navbar.

Our initial design for the homepage, however, did not include the navbar, as we thought it was to be more distracting. Instead we initially opted to just add buttons to the main home screen of all the different options. This was taken in inspiration from the previous design of the Ibeis app developed previously, as shown in Figure 3.

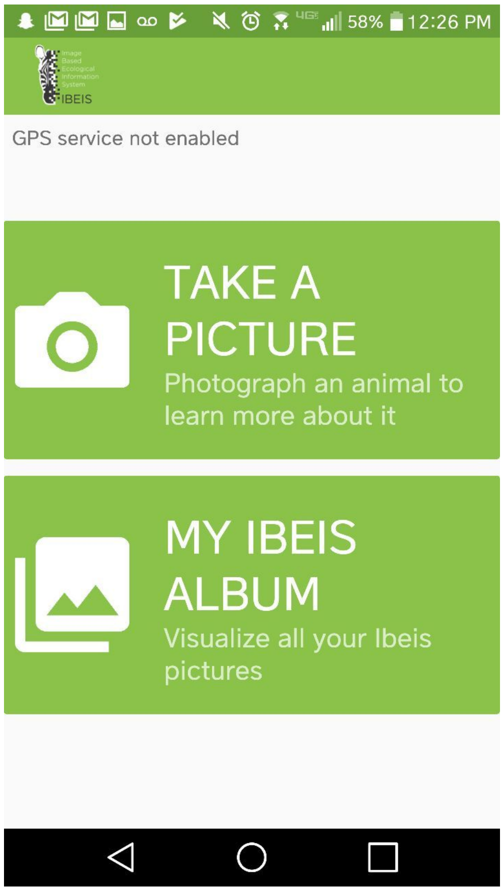
58%
12:26 PM
IBEIS
GPS service not enabled
TAKE A PICTURE
Photograph an animal to learn more about it
MY IBEIS ALBUM
Visualize all your Ibeis pictures

Figure 3

The first design was created in Adobe XD, as shown in Figure 4. It was, as seen, mirror the homepage while adding some more visual aesthetic to the user interface. This would expand upon the current design, which is rather bland, and would add the logo of the company (important for marketing), a short description of how the app experience is going to be, along with basic instructions on how to the use the app. The purpose of this design was to make it easy for any individual to pick up and use right away, appealing to the public rather than only a small group of expert individuals. Along with this the current location can be seen as an indicator of where the individual is, as well as letting the user know that their location services are turned on.

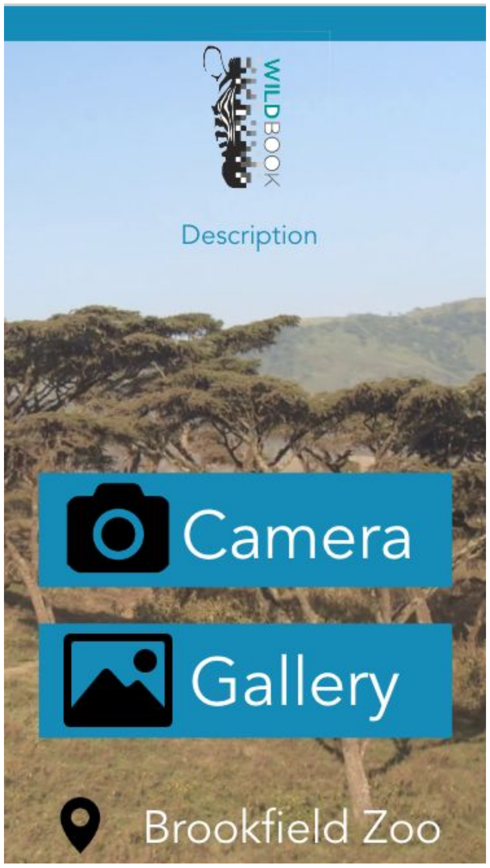

Figure 4

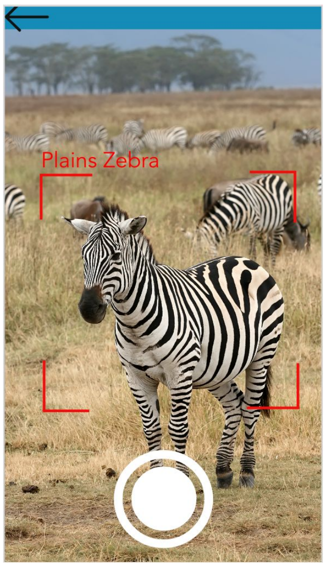

Figure 5

The end result of the camera is to have a live image recognition software run during the duration while the camera is on. As shown in Figure 5, our design for this was to include a small box with a label around the recognized individual animal in order to engage the audience in live time, and if so, even have the specific individual recognized for certain animals, including the name and age, all in live time. This would allow the user to gain information quickly, and if he/she were to need even more information, then taking a picture would provide that, which will be explained more in depth later.

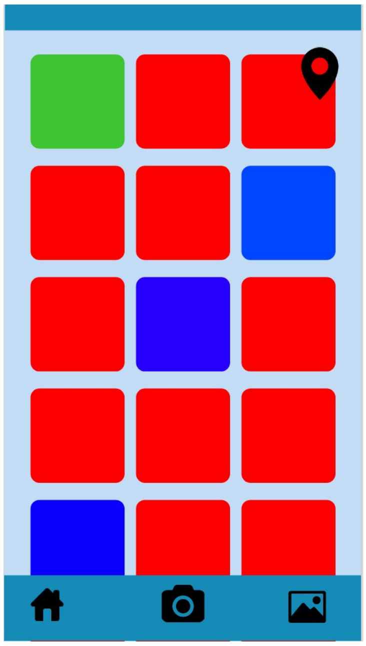

Figure 6

Figure 6 illustrates the initial design for the gallery feature of the application. Each tile represents a placeholder for an image, so that the user can quickly scroll through and view each of the pictures they have taken. It would allow for a quick and easy access to go over the experience. Also implemented is a toggle to switch to location based gallery, shown in Figure 7; this displays a map and a pin of where each image in the gallery was taken. This allows to see the variety in animals, as well enables the user to revisit the specific animals and connect with their trip a little better. Of course for future use in the wild this feature could be used to document and track where the animals were spotted.

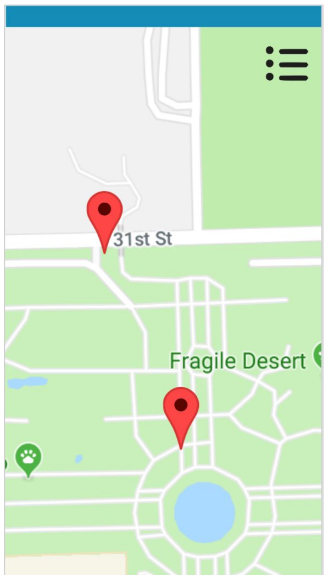

Figure 7

The last part of the design was the image detail page, which would be automatically generated for each image taken by the camera. This displays the full image for better viewing of detail, along with more info for the specific animal or the specific individual that the picture was about. The description, along with the details would allow more of a user to zoo direct interaction, as zookeepers could update the animal information on a daily/weekly basis, and thus everyone in the zoo using the app could scan the animal to be updated about the animal's most recent activity and information. The basic design idea is shown in Figure 8.

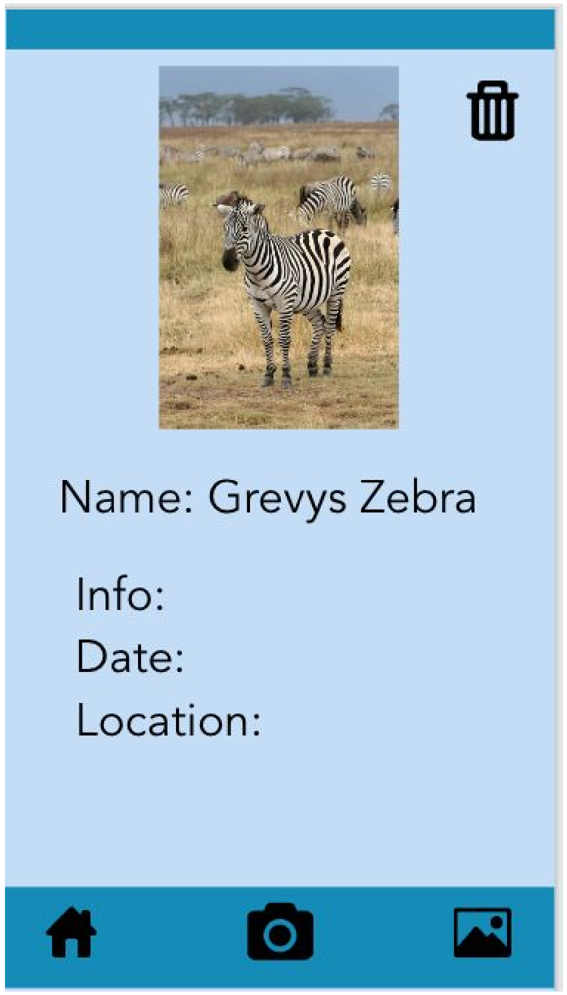

Figure 8

Overall, the initial design was meant to be more of a skeleton structure aiming to be a more malleable layout, being able to add, remove or change parts very easily and quickly if necessary. This modular design was aimed to be useful for all, and though rather simple, it would be effective in its purpose.

The Project

Rather than dive right into the details, the team started off this project with the basics, creating a basic layout structure first, getting used to the language and environment of Android application writing and starting out a more general approach compared to getting into the specific features of the application; adding buttons and images to the user interface layout, and creating these simple tasks as a setup to get into more complex functions. Once that was created, the team dove into more of the functionalities, starting off by attempting to implement a button to activate the location as well as the camera.

One difference between previous versions of the Android SDK (Software Development Kit) and the more recent ones is because of the European Union's updated laws surrounding privacy, the permissions requested and granted by the user and the application have changed. Previously, a simple acceptance of use of external services of the mobile device, such as access to location services, camera, flash, and even other hardware such as reading and writing of internal memory, would have sufficed in the past. The team had to quickly adapt to the updated policy of having to request twice, once at the initial start of the application, and once again at the initial startup of each feature. For example, at the initial startup, our application would ask for permissions to access the current location data of the user, along with the camera. However, when actually requesting the location data for the first time, another request would have to be made in order to ensure that little to no malpractice was being done by potentially abusing the application and reading and writing data of the user when it was not necessary. This was one of the changes that we, as a team, recognized would have to be implemented for all future requests, along with realizing the fact that if in the future more SDK requirements would be updated, then it would be necessary to update the application as well in order for it to have proper use. Many other developers have deemed this functionality of requesting both at initial startup and initial use rather unnecessary; while it is taxing to account for the in the development process, it does pay off in the fact that the user has their data protected, along with the fact they can entrust the developers a little bit more to not use their device for ill intentioned purposes. Either way, Android SDK forces everyone to use it as an industry standard, so having to implement it is inevitable.

Our first proper "build" included some of the basic buttons and service requests, along with displaying a null image on the screen, as shown in Figure 9. Although to the eye this may not have looked like much, we mainly focused on the back-end development of the application before polishing the user interface for public use.

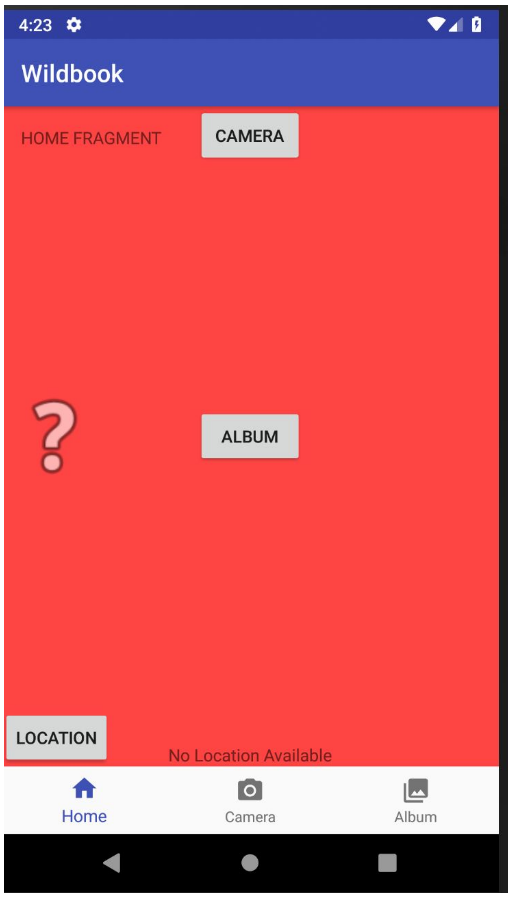

Figure 9

Along with the user interface implementation and integration with the back end, we ran into an unforeseen issue of using Android activities or Android fragments. When using the navbar, it is a good practice to keep the navbar constant when switching between different tabs in navbar, while simply changing what is displayed in the main screen. Fragments are, in a sense, a screen within a screen, allowing the contents of the box to be changed while everything else remains the same. This can be illustrated in both images in Figure 10. This allows for flawless transitions into the next tab, even though it may have trouble loading. Activities, however, involve the whole screen. This means that every time a tab is switched in the navbar, the whole screen would have to reload to display the new information.

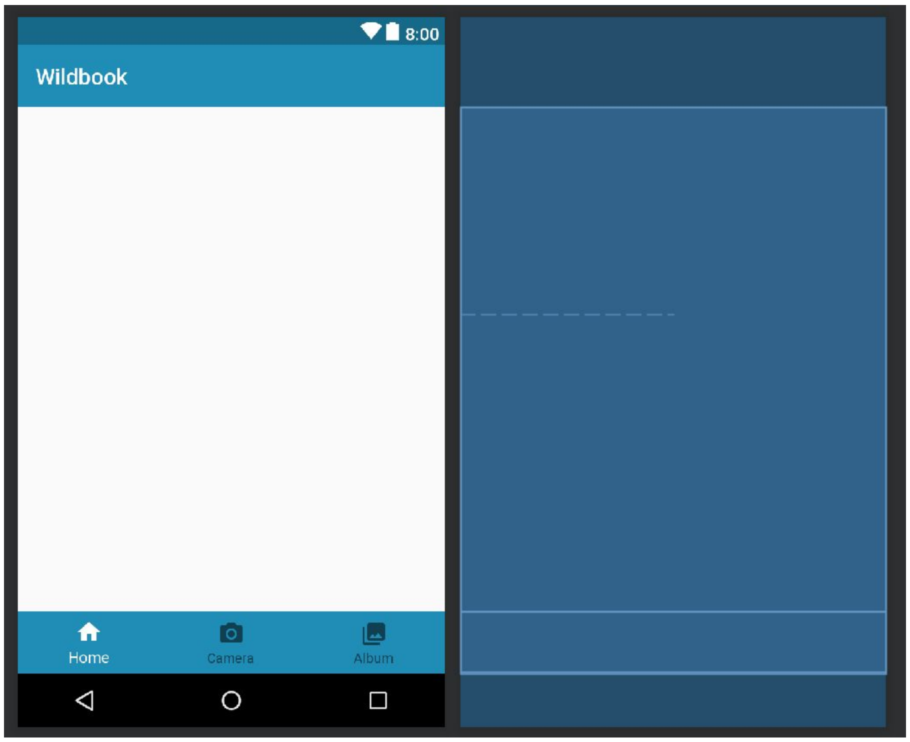

Figure 10

The team discussed this and weighed out the pros and cons of using fragments versus activities. On one hand, fragments are extremely useful because they allow continuity and keeping the flow of the user interaction smooth. However, fragments require running multiple processes, which takes up more processing power of the computer, as you are running a "screen within a screen" managing both the inside and the outside displays. Ultimately the decision was made to implement fragments into the application, again referring back to the application design of many more high budget apps on the market, even such as the original ESPN app which was referred to in the past, as shown in Figure 2. Having adopted this idea, much more planning of memory and processing usage was needed in order to make the run more effectively and efficiently.

One method of increasing efficiency and tightening up code was in the process of capturing the images taken by the in-app camera and storing them correctly. It was learned that the Android operating system works that if needed to take and save a picture, the picture saving method is done so in opening the camera, and not of taking the actual picture. This led to the issue of many null files, of the proper name but no picture, being created whenever the user would open up the in app picture but would not take an actual picture, and instead back out or quit the application. Although the files that were being created were of very small storage space, the team realized that on a large scale this would lead to many issues down the line. In doing so, we decided to implement a checking system that would delete the file if the file size was below a certain threshold of size. Yes, every time the camera app would open a file would be created, but if it was not filled with the picture, it would be deleted before many null files would stack up and create bigger problems.

Because Java is a dynamic language, it is very hard to manage the current memory storage of the application, and each application is only granted a small amount of space by the operating system. Therefore we quickly learned that memory would need to be optimized instead of having to wait for the garbage collector to randomly clean up unused storage during the use of the app. A static language allows the developer to have full control on where and what memory is allocated and deallocated. However a dynamic language assigns the compiler to take care of this process. While it is useful not having to worry about pointers and addresses to memory, unused objects in the development process may take a while to clear due to the uncertainty of when the uncontrolled garbage collector will free up the unnecessary objects. Therefore the team had to be extremely clean and cut in using and maximize the small amount of memory storage that was allowed.

This issue brought itself up in the displaying of the gallery. When trying to display all the images as tiles, we were pulling the high quality, often times 4k resolution images directly and displaying them as a tile thumbnail. This unnecessary high quality resolution of displaying possibly hundreds of images, each 4k resolution or possibly more would result in the almost instantaneous wastage of memory. The app would crash immediately. Therefore at first we were only able to display one image tile at a time at the 4k resolution, as shown in Figure 11.

Although not implemented yet, the team created a plan to be able to accurately display all of the image thumbnails at once. First and foremost, the program would need to pull each image, and at a lower resolution, crop it accordingly. Along with that, delete the original high quality image from the easy to access memory, and instead only bring up the high quality image when displaying the full version in the information window. Along with simply the image quality reduction, a method to pull up all the images at once instead of having to individually hunt for each image in the memory location and pull up a new search each time. This combined with a possible Image Class in order to organize the data more properly would most likely needed to be implemented in the future to optimize the performance.

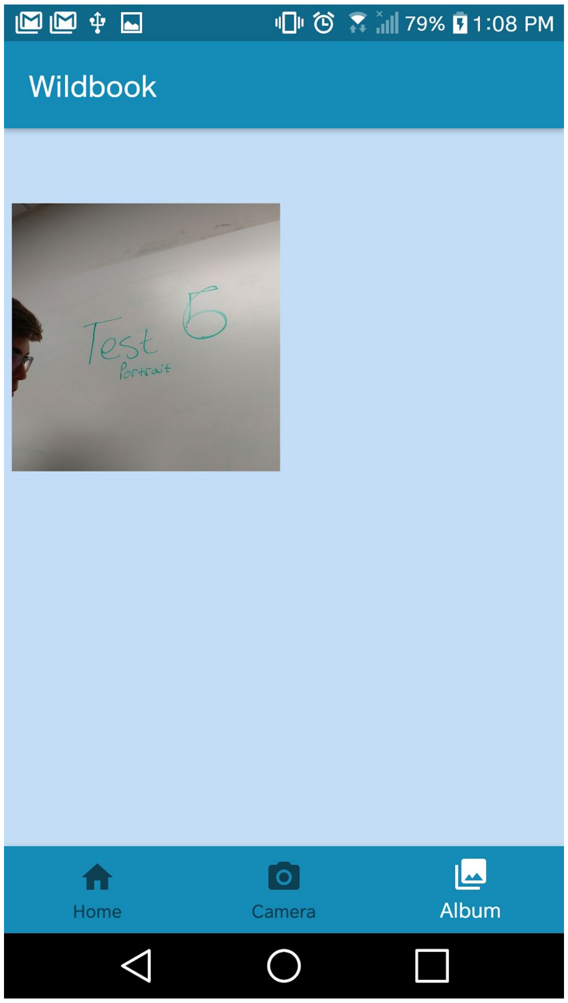

Figure 11

All in all, after completion of the main parts, a color scheme was implemented and aesthetics were used to improve the design and overall feel of the application experience, including themes and minor updates like highlighting the color of the current tab on the navbar. The home page, while not fully complete, is as shown in Figure 12, is far more pleasing to the eye than what was initially developed.

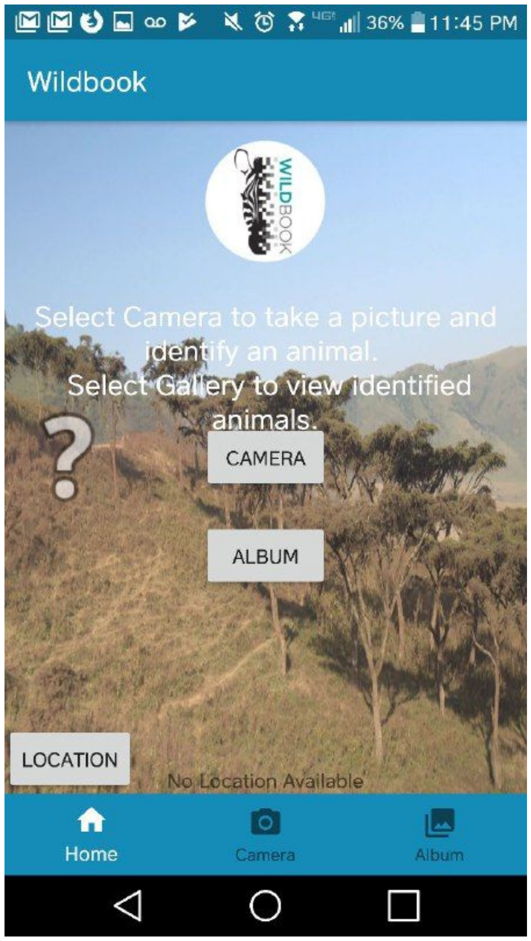
36% 11:45 PM
Wildbook
WILDBOOK
Select Camera to take a picture and identify an animal.
Select Gallery to view identified animals.
?
CAMERA
ALBUM
LOCATION
No Location Available
Home
Camera
Album

Figure 12

Future Plans

As of the end of August 2018, the application development process has more than a solid foundation, but still has ways to go before able to release to the public. The development team had discussed this, and has come up with both short term and long terms goals in order to truly make this the best that it can be.

Starting off what can be completed in the near future, the design and user interface could be improved with an upgrade, customizing buttons and the the icons, as well as matching the proper color text to go along with the applications current theme. Moving deeper in, the application could really extend its long term functionality with better data structures embedded in the app, for example, with Photo/Image Classes, along with a Gallery Class and an Information Page Class to make it easier to read and change the current code that is present. The information page itself needs to be implemented as well. Currently when the image in the gallery is clicked, a larger version of the image simply appears. It should not be too hard to implement, but this is one of the short term goals that can be accomplished sooner than later. Lastly, the development team thought it would be best to be able to crop an image taken with a camera before it is saved to the memory, as it would allow more accurate image recognition, and would save a lot of space, along with eliminate unnecessary bad pictures filled with non-animal space.

The image recognition is probably the priority long term goal, as that will be the defining feature that would set apart this application from the rest. Long term ambitions also include making this application almost as a form of social media, allowing users to view other individuals images from the same zoo, or of the same animal. This could be used to document the growth and progress of the animals/species as well. Along with this, zoo keepers and other researches could regularly update the information details page about the specific animals in order to engage the users to another level. For example, a user could simply snap a picture of a zebra and the application would let them know what it ate for breakfast this morning.

Conclusion

All in all, this experience was a very fruitful one, to say the least, for the whole development team. A great start on this application was achieved, with a good application design process of design, implementation, and looking for future improvements. The team worked well together, making sure to communicate and collaborate on all ideas and issues, as well as dividing up certain tasks when necessary in order to be more efficient. We have certainly taken a lot from this experience, as well as given back in our development of the application, both about development, and the development process as well. In conclusion, the goal is clear, the foundation is strong, and the application has made significant progress. We wanted to thank our professor and all the lab members for helping to make this process easy and fun for all of us, and we hope to be able to see this project come to a successful completion in the future.