Bee-cology:
A Citizen Science Mobile App

A Major Qualifying Project submitted to the faculty at

Worcester Polytechnic Institute

in partial fulfilment of the requirements for the

Degree of Bachelor of Science

By:

Jacob A. Moon

Ziyang Yu

September 6, 2017

Professor Carolina Ruiz, Project Advisor

Department of Computer Science,

Bioinformatics and Computational Biology Program, WPI

Professor Robert J. Gegear, Project Co-advisor

Department of Biology and Biotechnology,

Bioinformatics and Computational Biology Program, WPI

Professor Elizabeth F. Ryder, Project Co-advisor

Department of Biology and Biotechnology,
Bioinformatics and Computational Biology Program, WPI

Table of Contents

Table of Figures

Abstract

The impending depletion of local pollinators is threatening the balance and overall biodiversity throughout the global ecosystem that is earth. Properly combating this downward trend requires a massive amount of data, which was previously uncollectable through conventional means. Therefore, this MQP worked to facilitate a modern approach to data collection on local pollinator species, through the development of a mobile application for public usage on Android devices. This project builds on previous work (Jackson, 2017) using the basic structure provided to ensure a stable and functional application to be distributed for public usage. Following this rationale, the goal of this project was the improvement of its predecessor. Utilizing Java programming required through Android Studio, we made the following improvements to the previous app: 1) improved functionality and accuracy of bee identification and information collection process, 2) implemented functionality for transmitting data from local storage to a central database, and 3) improved user experience and application aesthetics. Through these improvements, we have produced a functional application for the purpose of crowdsourcing of pollinator data collection and retrieval for ecological researchers. In the future, other project groups will work towards expanding and improving this application for public use.

Acknowledgements

We would like to acknowledge and thank everyone who provided a tremendous amount of information and assistance for this project.

First, we would like to acknowledge our project advisors from Worcester Polytechnic Institute:

- **Professor Carolina Ruiz** from the Department of Computer Science for giving us vital technical feedback and advice about the application.
- **Professor Robert Gegear** from the Department of Biology and Biotechnology for giving us important information on bumblebee species and design constraints for the new version of the application.
- **Professor Elizabeth Ryder** from the Department of Biology and Biotechnology for providing relevant user feedback about the application and for providing advice on structure of the documentation.

We would also like to thank all the other students and teams who provided essential assistance to this project:

- **Jackson Oliva** from the previous Bee-cology MQP for building the original model of the application and giving us insights about our goals for accomplishing this project.
- **Xiaojun (Susan) Wang** for establishing and maintaining the central database of the Bee-cology project and helping us create the communication process between the new application and the central database.
- **Kenedi Heather**, **Rachel Murphy**, and **Devin Stevens** from the previous Bee-cology IQP for providing bumblebee color pattern graphics and the original bee identification process model.
- **Sam Coache** from the current Bee-cology IQP for providing insights on the new flower identification model.

1. Introduction

One important global problem is the ongoing decline of wild bumblebees and other important insect pollinators. Although the cause of these declines is currently unknown, invasive species, habitat loss, pesticide usage, transgenic crops, and environmental pollution are all thought to be significant contributing factors (The National Academies Press, 2007). To develop effective pollinator conservation and restoration strategies, a critical first step is gathering large amounts of data on ecological habits and needs of each pollinator species. Important ecological data to collect includes: nesting habitat, floral preferences, and species phenology. This information could in turn be used to inform the ever-concerned public and garner funding to support endangered populations, on a regional and a local basis. With enough information about each individual pollinator species, we will gain greater insight into consequences of pollinator decline for ecosystem function and biodiversity.

To initiate the process of collecting ecological data on wild pollinators, previous MQP work resulted in the development of a Citizen Science Android Application for bumblebee pollinators. This application was designed to collect data about bumblebee-plant interactions in Massachusetts. The application enables users to follow a guided identification tree, to identify the bumblebee species and the floral species that they were visiting. The user could then save the logs of collected bees on their mobile device.

While this version of the app helped educate the public about bees, it did not transfer information to the Bee-cology database. In addition, the app had problems with design, structure, and was not fully operational. Our MQP project focused on further developing the app so that it can be used by citizens participating in the Bee-cology research project. To do so, we aimed to accomplish the following three goals: 1) improve functionality and accuracy of bee identification and information collection process, 2) implement functionality for transmitting data from local storage to central database, and 3) improve user experience and application aesthetics. The remainder of this document covers how we reached these goals, detailing all of our major updates, minor updates, and the design and implementation of novel features, in reference to the aforementioned application failures.

2. Background

The unprecedented worldwide decline of wild pollinators over the past two decades poses an imminent threat to global biodiversity and food security. In Massachusetts alone, one half of our native bumblebee species are in decline, with one recently listed as an endangered species and many soon to follow. Previous researchers (Oliva, 2017; Heather, Murphy, and Stevens, 2017) worked on developing the the most of effective way of educating and engaging the public in helping the survivability of their local pollinators. Through their research it was determined that this could be accomplished through a Citizen Science mobile application (Olivia, 2017). Therefore, this project will focus on how their research was continued and implemented.

2.1 Bee Identification and Information Collection

To continue work on this application, new additional research must be completed to help add and improve features. For this application, a significant portion is the ability to correctly identify a multitude of local pollinator species, to garner insight into their biodiversity. While identifying the pollinator is important, equally relevant is all the other information present e.g. the flower, location, time, and bee behavior (*Williams,2014)*. This following paragraph will provide some basic information into bee identification for this project's discussion, however, more in depth research can be found in the previous MQP (Olivia, 2017) and IQP (Heather, Murphy, and Stevens, 2017) documents.

To identify any bumblebee species, the observer must understand the three portions of a bee, the head, thorax, and abdomen. These partitions can be seen labelled in Figure 1.

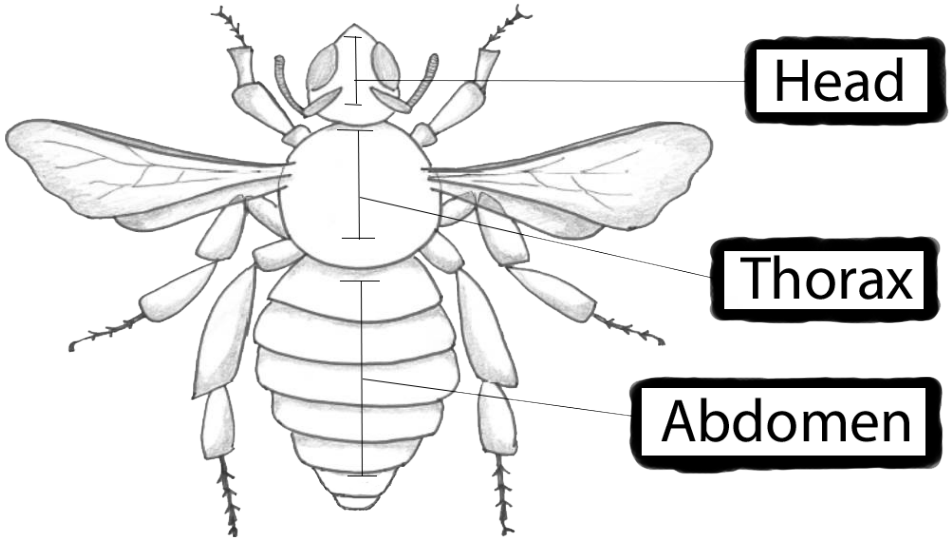


Figure 1: Basic anatomical model of bumblebee

The different color patterns in these three regions work to identify any bumblebee. Cartoon images of each bumblebee coloration pattern were created by a previous project team (Heather, Murphy, and Stevens, 2017) to help identification and these images can be found Appendix C. In addition to identifying the species, the coloration on the head (and occasionally the thorax) can identify the sex of the bumblebee. However, while the coloration pattern can be distinct the small size and quick movement results in a significant challenge to the layperson.

While it is important to collect the species of bumblebee, it is also pertinent to identify the behavior of the bumblebee being tracked. Bumblebees can be spotted while participating in two distinct activities collecting either pollen or nectar (*Goulson, 2012)*. Whether a bumblebee colony is collecting pollen or nectar can tell us a lot about the state of the hive (*Goulson, 2012)*. This is because of the function of these two resources, pollen is used as protein for the young, and nectar as energy for workers (*Goulson, 2012)*. In the early spring a queen will first collect lots of nectar to gather her strength, and then begin collecting pollen to grow the hive. A bee collecting pollen can be identified by a presence of a buzzing noise and the bee walking along the stamen of the flower (*Goulson, 2012)*. While a bumblebee collecting nectar will be seen going into the flower and using its tongue to infiltrate the plants nectaries. In contrast, a bumblebee collecting pollen can be seen shaking or buzzing on the anthers of the plants.

Apart from the species and behavior, it is vital to recognize which floral species the pollinators are interacting with. However, flower identification represents an expansive problem on its own, therefore, it is simpler to collect a few unambiguous data points such as the color and shape. This data can be compared with known information about bees such as tongue length, to speculate on the preferences of each pollinator. Discovering which plants pollinators prefer is essential to recommending pollinator friendly plants to the public.

2.2 Citizen Science

This application has been created and implemented to facilitate a Citizen Science project. This means that the application is designed to be utilized by laypeople and to effectively crowd source scientific data (*What is citizen science?.2017)*. With this format there are many positive and negative aspects.

The positive aspects are numerous and include the ease of data collection, cost efficiency, quantity of data, and public engagement (*What is citizen science?.2017)*. These benefits are generated from the very nature of citizen science. By expanding data collection to the individual rather than the organization, a large quantity of free labor is accrued (Bar & Maheswaran, 2014). Simultaneously, the user must be guided through multiple simplistic tasks each of which builds towards pollinator identification, ensuring that the application is easy and quick to use. This project will hope to incentivize users through easy use and the ability to view their impact towards scientific research.

However, relying on the public has many drawbacks as well including, difficulty recruiting and retaining users, inaccurate data submission, and possible variability in user submissions (Bar & Maheswaran, 2014). The nature of the project requires the use of the public to collect this information and as such, it is necessary to recruit support for this project. After recruiting an individual it then becomes pertinent to train them in proper use and foster a desire to continue using our platform. The application should be easy to use, thereby limiting the amount of time it takes to train a new user. Lastly, user retention is another large problem which will take time and energy of the institution to foster and combat the impending decline of use (Bar & Maheswaran, 2014).

To foster the positive and limit the negatives of this format is imperative for this application to have an easy and simplistic design that can keep track of users and the data they submit. To make the application straightforward and easy requires researching into functional and aesthetic issues within the previous application. Additionally, new features and screens will need to be implemented such as, tracking users and allowing data submission.

2.3 Mobile Application Development

Following the description of how this application hopes to achieve broad scientific goals, it still must live within definable technical constraints. To this end, the following section will detail the problems with the previous version of the app, and the constraints we face as we attempt to add and improve features and functionality.

2.3.1 Previous Application

The previous project had broken down their goals into three distinct categories: "the ability to log a bee, the ability to look up information about local bumblebee species, and the ability to view past submitted logs" (Jackson, 2017). However, while these goals served as a necessary starting point, they would have to be revised to continue the application. For this project, these goals have been reformatted as follows: (A) improving functionality and accuracy of bee identification and information collection process, (B) implement functionality for transmitting data from local storage to central database, (C) improving user experience and application aesthetics. The following paragraphs outline how the previous project completed these goals.

A. The previous project utilized a two step identification process. The identification was done first via a trichotomous decision, accomplished by asking the primary color of the thorax denoted by three options: "mostly black", "half black/half yellow", "mostly yellow". Then the user was confronted with a traditional dichotomous decision tree; in this case the application would ask yes/no questions, until the user identified one of the possible bees. A portion of this decision tree can be seen in Figure 2, while the entire tree can be viewed in Appendix D.

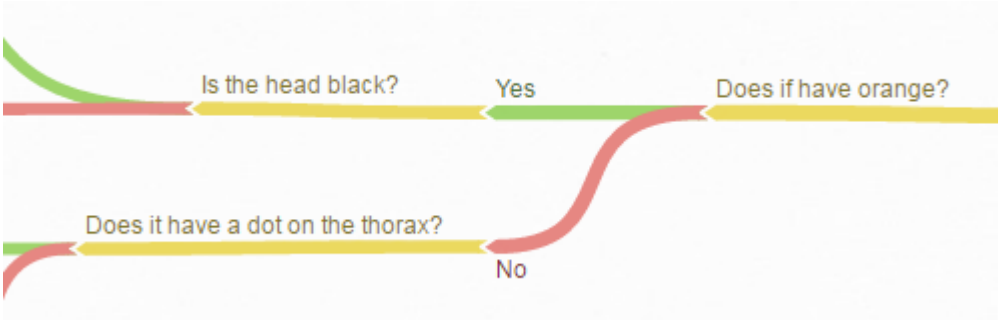


Figure 2: Portion of original identification tree. This portion of the question tree shows the basic structure of the identification process, where the user is asked a series of yes/no questions. Taken from (Heather, Murphy, and Stevens, 2017).

While this is an effective method for identification, it has a few drawbacks. Primarily, not all branches of the tree would end at an identified bee. Secondly, if it was desirable to add new bees, the entire tree may need to be reformatted to function. Lastly, if the user happened to make a mistake, then the application would give a them a prompt saying the bee couldn't be identified and then redirect them to the start of the identification process. This did not alert the user to errors or even limit the options of where they could have been mistaken.

B. The previous project had made it a priority to store all information from each log into a local database on the mobile phone. The local database was created through using SQLite, which is free, open source program which offers an easy and efficient way to store multiple data tables (About SQLite, 2017). This feature served as a way for the user to store data independent of internet connection, freeing them to take multiple logs in remote locations. However, there was no way for the user to submit these logs into the central database. Another issue was that images would not be stored in the mobile device's internal memory, but instead as a bitmap in the mobile device's RAM.

C. The previous project had a minimalistic structure as to the quantity of information that was collected about the user and about the bees. Unfortunately, from this edition only the basic flower information and bee species was collected. In the grand scheme of things, this information only gives researchers the What, but neglects, the Who, When, and Where.

One major issue presented by this, is that any user could have theoretically, submitted large quantities of false data, and there would be no way to invalidate or identify the source. Another major issue presented in this application was that the user only had the ability to identify bees through taking a picture or video within the application. This limited the user to utilizing the application for one pollinator at time rather than photographing multiple pollinators and identifying them later. This problem could be solved by allowing the user to import previously taken photos.

2.3.2 Building and Testing the Application

To construct this application, we had to utilize a variety of technical resources for individual programing, collaborative work, and testing of the application. To work on individual programming, the application, this project utilized Android Studio. "Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA" (Meet Android Studio, 2017). This was recommended by the previous project team and enabled this project to seamlessly work on both the frontend and backend of the application. Android Studio enables the application to be programmed and written in Java 8 for all Android mobile phones with SDK API 23 or higher. This SDK level was chosen to enable a wider variety of features and due to its compatibility with most modern devices. Likewise, Android Studio offers application testing through multiple mobile device emulators (Meet Android Studio, 2017). Android Studio also enabled this project to integrate with multiple collaborators via GitLab (Meet Android Studio, 2017). GitLab is a program utilized for multiple programmers to work on the same application separately and effortlessly connect their work. Lastly, this project will utilize many mobile devices for testing, from associates and advisors. However, this project will conduct the most rigorous quality assurance with a Nexus 5X to ensure stable use on mobile devices.

2.3.3 Database Communication

To collect the information about the relevant bee species, it is a necessity to connect the mobile applications to a central database. To accomplish this goal the application must store all data

locally, and communicate with a web service application, which will then submit the information to a central server. The previous project team created the local database for storing relevant information about each log. Simultaneously, a graduate project has been working to create the webservice and the central database. The web service is a RESTful API that utilizes the Swagger (OpenAPI) specification (*Swagger Specification, 2017*). For this application to communicate it will utilize GET and POST requests to the web service (*Swagger Specification, 2017*). These requests work by sending or receiving JavaScript Object Notation (JSON) files. The GET requests will be used for retrieving information from the database, such as previously submitted logs. The application will utilize POST requests to submit data to the database, such as a recently completed log (*Swagger Specification, 2017*). To complete these transactions, the application will require a stable internet and must have the data in the appropriate format.

2.3.4 The User Interface

While fixing functionality and stability issues, is vital, another pertinent aspect of this project was to amend the aesthetics and user experience. This is due to the nature of a citizen science project requiring a simple and rewarding design that can simultaneously limit confusion. A significant way to reduce confusion is to improve the method for identifying pollinators. Confusion could arise if the user made any mistakes when answering questions in the dichotomous decision tree. Another drawback in the application's user interface was the inability to properly format screens on a variety of mobile devices. An advantage for utilizing android is the large number of users; however, a common hindrance is lack of a standard for device manufacturers. This lead to many issues with both the frontend and backend development of the application. This resulted from a lack of planned variability in the aesthetic aspects of the application. This means that all images, text, buttons, and widgets are set to specific pixel sizes, rather than changing dynamically in reference to the screen size and base user settings. The user interface had redundancies which could lead to user confusion, such as a back button which was not needed, since it is a default found on all android devices. Lastly, some of the images and descriptions of bees were not very helpful to the user, such as the original bee images displayed during the identification process.

2.4 Conclusion

The next chapter of this paper is the Methodology, which will document the changes that were implemented. The previous chapter outlined the research necessary to discover the areas requiring updates. All changes implemented focused around the following three goals: improving functionality and accuracy of bee identification and information collection process, implementing functionality for transmitting data from local storage to central database, and improving user experience and application aesthetics. Improving accuracy and functionality of the pollinator identification process entailed revamping the current question tree and collecting additional information. To enable the transmission of data from local storage to the central database required collaborative work with other researchers and the addition of novel features. Lastly, improving user experience entailed focusing on user feedback and removing bugs from the application. The full explanation of how these goals were implemented is explained below.

3. Methodology

This section outlines design and functionality problems with the previous version of the App that we needed to solve in order to meet our project goals.

3.1 Problems with general App development strategy and quality assurance

The development of the new version of the application had three goals. First, we wanted to improve user experience, therefore gathering user feedback was pertinent part of development process. During the development process any updates to be made were vetted before work began and after completion. Secondly, we wanted to maintain a semblance of modularity to make future updates easier. For example, based on user feedback it was important to replace all of the images displayed in the pollinator species information page. To facilitate an easier replacement in the future, instead of storing all information in the layout level of the application as was done in the previous version, it was stored in the local database and extracted by the layout for use. At the same time, to raise the potential of future developments of the application, we chose to develop the new version of App with JDK 1.8. Third, we wanted to ensure that any feature to be added or improved during the development process resulted in stable usage across multiple android phone models. To accomplish these goals required certain SDK levels and use of the Google Play Service, to be consistent throughout the application. During the development process, we used the most updated version of Android Studio (V2.3.3) as our developing platform and GitLab as our working repository. To ensure the app is compatible with a high number of devices and lacks any functional defects, we used multiple methods and devices to test the app before releasing each feature. First, we used an emulator provided by Android Studio. This allowed us to have initial views and functionality tests on every newly updated feature. We used the Nexus 5 with API 25 and Galaxy Nexus with API 25 as our virtual testing devices. However, because an emulator is a software that simulates the operating system of an android device, it does not contain any sensors, such as a GPS or camera. Therefore, some features were not testable, for example, taking a video from the application or requesting the current location of the user. Following these limitations, it became necessary to utilize the application on real android devices to finish our debug process and usability

testing. Our main testing device was the Nexus 5X with API 25. We also used a Motorola Moto X with Android Marshmallow 6.0, LG G4 and Samsung Galaxy S8 to do the usability test and ensure the application was functional on devices with different manufacturers and android versions.

3.2 Major technical problems with the previous App version

3.2.1 Identification Method

The previous version of the application identified pollinators through the utilization of a dichotomous question tree. However, this method had severe limitations and problems. Primarily, the question tree had predefined paths toward each identification. Having hard-coded paths would have required reformatting the entire tree, whenever a change or addition would be made, such as the inclusion of gender variants. Secondly, this method lacked any tolerance for errors within the question tree, because the user would not know at which fork they selected an incorrect answer. Tertiarily, the questions were focused on coloration and the application displayed the user with a cartoon image of the possible pollinator color pattern, to compare at each stage, which was an unnecessary redundancy.

The new version of the application had to find a better method of identification to improve the user experience and ensure a higher rate of accuracy. To accomplish this, the dichotomous question tree was removed and replaced with a polyclave key. A polyclave key works by allowing users to select from multiple options for multiple traits, and then querying for a result with matching fields. By letting the user select from multiple options drastically reduces the number of questions and therefore, ratio of error. The new version of the application, had this polyclave key created via utilizing the cartoon images of the pollinator coloration patterns. The current method allows the users to chose between different coloration patterns for the head, thorax, and abdomen, before querying a local database for the matching pollinator. To implement this a local database had to be created wherein all of the species and gender variants, are stored along with their respective coloration patterns. Additionally, the cartoon images had to be updated and the ability for a user to select an “unknown” pollinator was added.

3.2.2 Video Capture

The previous version of the application contained the ability for the user to take a video and then capture a single frame from the video. This feature is necessary so that users can take videos of quickly moving bees and then find a proper image from the video. Proper image meaning one in which the head, thorax, and abdomen coloration patterns are clearly visible. However, the previous application's video capture method was not fulfilling this goal, because the video was not a video, but instead a gif that played in real time and could not??? be paused. The current application solves this issue in two ways, slowing down the video prior to playing, and the addition of a scroll bar, which allows users to choose the frame being displayed. To implement these solutions required updating the application's SDK version to 23, which allows developers to set the displaying speed for the media player in the application (Playback Params, 2017).

3.2.3 Backend Image/Video Handling Method

The previous application had consistent issues with the way it handled incoming images and videos. In the previous application, images were stored as bitmaps in the mobile device's RAM, and these bitmaps were then passed from screen to screen. Additionally, videos were stored via similar method, however, videos were only temporarily stored during the editing process. This limited the user to a single screen capture, before the video was deleted. Therefore, this was forcing the user to take another video, in the event of a mistake. Handling images and videos in this way resulted in three concerns; users lacked the ability to go back to a previous screen, the images were not being submitted to the central database, and image quality was limited by the mobile device's RAM. The current version had to resolve these issues by developing a new system for image handling, that did not compromise user experience. The solution to this problem was by having these files saved locally and only transferring the image path, rather than the entire image bitmap, therefore reducing the work imposed on the device. To accomplish this, the current application had to create temporary image and video files on the mobile device, that could be deleted locally after database submission.

3.3 Minor problems with the previous App version

3.3.1 Spacing Issues

One of the complaints provided by users of the previous application is the text in the application was occasionally being cropped off of the screens. This issue usually appeared if the user installed the application on a phone with low screen resolution or changed the font size setting of the phone to maximum. The previous application had the majority of the text, images, and widgets in each screen, set with static margins. The current application, has all of these objects placed dynamically, to reflect the settings of the current mobile device. To enable this, height and width of every widget and layout in the application had to be changed to encompass the varying size of content. For example, previously a scrollview's height was set to "500dp", a static value, however, now that scrollview is set to "wrap content", a dynamic value. Additionally, all widgets were reformatted to be wrapped inside a placeholder, instead of trying to multiple widgets in one screen directly. Through dynamic layout design the application became consistent and more convenient for future UI developments.

3.3.2 Bee Images

The previous application enabled users to compare the pollinator being logged with both real and cartoon images of the possible species match. However, the images provided the user with little guidance towards identification. This was due to inaccuracies, in the cartoons, and poor lighting, in the field images. Therefore, all the cartoon images were reviewed by developers and experts, any wrong designs and coloration patterns were replaced by proper versions. The real pollinator pictures were taken, in darkness, of a collection of bumblebee samples in lab, resulting in most of images lacking a clear view of the abdomen, thorax, and head. Since these three regions are the main factors for the user to identify the species and gender of pollinator, these images were replaced. The current application features images of pollinators in proper lighting with clear representation of all body parts.

3.3.3 New Icon

Another issue that users had while interacting with the previous version of the application was the icon made it hard for users to locate the application on their phones. One reason that caused this issue was the icon design only occupied a quarter of the space allotted for the icon. Included in the previous icon was just a line of text which read “Bee-cology”, next to a small bee cartoon image. The current application features a larger icon, which fills the majority of the space allotted and is catching to the eye. To accomplish this, the new icon was re-designed to incorporate an easily identifiable nameplate, and inspiration from bees and flowers.

3.3.4 Removal of Back Button

The previous application featured a common redundancy, the addition of a back button on each screen. Based on this project’s research, “all Android device provide a Back button for back navigation” (Providing Proper Back Navigation, 2017). To remove this redundancy and improve the user experience the new version of the application had all back buttons removed. However, to accomplish this and keep the functionality of back navigation, each screens’ activity file had to incorporate an onBackPressed function. This function was designed to pass the necessary information to the desired screen, therefore allowing backwards navigation of the application.

3.3.5 Expert ID option

The presence of an expert ID process was integrated to facilitate expert users to log pollinators with minimal time requirements. In the previous version of application, the expert ID method contained an image for every species and links to detailed species information. However, considering the new version of application includes the field “gender” and multiple new species to the in the local database, created a few issues for users. First, users would’ve had to scroll through an extensive set of images for each pollinators and their respective coloration patterns, which can vary by gender, drastically increasing user search time. Secondly, this

extensive set of images had taken considerable amount of the mobile devices RAM, on older phones this or less powerful devices this caused the application to crash. Lastly, the previous presence of the pollinator images represented a redundancy in the application, the expert ID process is only for use by experts, who can identify a pollinator without visual aids. In response to these problems, the new application had to include a streamlined expert ID process, to increase user efficiency and reduce phone memory. This was accomplished through the removal of all images and replacing the scrollview with two drop down menus, one for species and the other for gender.

3.3.6 Screen Refresh Handling

Whenever the previous application was directing users to another screen, the new screen would consistently "roll up" from bottom of the screen. This was caused by a "refreshing" effect when users were interacting with the application. From the user feedback recieved, this effect was determined to be nauseating after long periods of use. To improve the user experience, the new version of application had to implement a transition-style guide on the application level of each screen. To accomplish this, a folder of animation guides had to included. These animation guides, provided the application the ability to apply a transitional style to each activity which changes screens. Whenever there had been a pending transition between screens an entry and exit guide was provided. In the current version of the application, the duration of these guides was set to zero, ensuring any transition is instant.

3.3.7 Additional Log Information

The previous application only collected basic information about each pollinator logged. Therefore, in order to increase the research value of each log, several new fields of information had to be implemented in the current version of the application. This project selected four separate fields to be included and improved in this application's release: pollinator species, pollinator gender, pollinator behavior, and flower identification. First, a new species was added to the the local database, *Bombus borealis*. This addition increased the number of pollinators that can be

identified, therefore, expanding the application's functionality. Second, the gender information of each species had to be added to the application. In this regard, each gender variant will be treated as separate type of pollinator within the identification process. Due to inherent gender roles in a pollinator population, this data inclusion has implicit value to researchers. Third, pollinator behavior had to be identified as either collecting nectar or collecting pollen, this was implemented through the addition of a drop down menu. This information was included to enable researchers to investigate the growth and health of the hive. Fourth, a new a text box was added as an option in the flower identification screen, this enables users to manually identify flowers, therefore, increasing application functionality.

3.3.8 App Procedure History

The previous version of the application had excluded the ability to save a screen's current history. This resulted in the the progress of the application being lost, if application usage was interrupted. For user's this became frustrating and represented a serious of hindrances to usage. The application had to be improved to allow users to "minimize" the application and continue their session later. To make this change for the new application, it was necessary to edit the AndroidManifest file, in which the information for each screen contained a field titled "noHistory" which was changed to from "true" to "false". This change ensured the current activity stack would not be closed when a user navigated away from the application, therefore, allowing the application to be resumed without loss.

3.4 Miscellaneous Problems with the previous App version

3.4.1 Importing Images

The previous application had no method for the users to import images. This problem was discovered via user feedback, wherein, the users were forced to take an image of and identify only one pollinator at time. This became frustrating for a user in the field surrounded by multiple pollinators, due to the length of time required to identify each species. Therefore, the current application had to include a method for importing pictures, to allow users to photograph and

identify independently. To implement this change required two updates to the application, mobile device permissions and proper image handling. First, the application required the necessary permissions to access the internal and external storage on the device. However, any selected image could only be opened and accessed, through this method, on the importing screen. To mitigate this issue, images had to stored in a temporary file in the local database, and handled as any other image.

3.4.2 Location/TimeStamp

The previous application lacked the ability to collect any location or time data about the pollinator being logged. However, a necessary aspect of pollinator research is knowing the region and date when each specimen was documented. Therefore, the new version of application had to record these details in each log. However, the location and date would have to be acquired through two separate methods, depending on how the image was received by the log,

Primarily, if a user was utilizing the default image capture method inside the application, the current location and date information would be captured from the mobile device. However, this feature would require user permission to access device hardware sensors, such as GPS. Secondarily, if a user imported a photo from the device's photo gallery, then the application would have to require a manual input for location and date. Manual input is required in this case because importing implies the user is identifying pollinators either offsite or at another time. To enable manual input of location and date, required a way for user's efficiently select both. Therefore, date was chosen to be selected via a scrolling menu, where the starting date is the current one. For location, it became necessary to have two different input methods. First the application had to utilize a google location picker, so that the user could manually pinpoint the location of the pollinator on a map. Secondly, all of the locations recorded in this manner had to be stored on the mobile device. The stored locations are used in the creation of a drop down menu, sorted by date inputted, from which the user can select the desired location.

3.4.3 User Log in

The previous application lacked a method for identifying and tracking users. This compromised the accuracy and accountability of every submitted log. Without user identity, researchers would not be able to exclude users with invalid data, or communicate with users about the data collected. One method to implement this was to create a user login screen, which would collect the user's email address. Based on the guidance from user feedback, the email address was decided upon for the following reasons: ease of communication, not generally considered sensitive information, already unique, its usability as an identifier on multiple platforms. To properly implement this solution created two criteria accuracy and consistency of the user's inputted email address. To ensure accuracy, any inputted email address undergoes basic validation. To ensure consistently on at least one device, the email had to be recorded locally, to prevent the user from changing it and enabling it for utilization with data submission.

3.4.4 Submission to Central Database

The previous application did not have a method for user collected information to be submitted to researchers. Therefore, a major feature which had to be implemented was the ability to submit locally stored logs to the central database. The necessary steps for the implementation of this feature are broken down as follows: satisfying precautionary requirements, data submission process, and data preparation method.

To allow the process of data submission, two precautionary requirements were taken to ensure proper implementation. First, the application had to test if the user has an active internet connection, preferably Wi-Fi, due to the possibility of large data transactions. Secondly, the application had to process the submission request on a background thread of the mobile device. This was a necessity because when the mobile device is processing any request it is constantly refreshing the screen. Therefore, if the device's main thread is entangled processing a multi-second request, then the application can crash, skip frames, or appear frozen to the user. This is

not a significant issue for an individual log, except if the user intends to submit more than a few logs at once.

The current application had to implement a consistent data submission process. This was accomplished through the utilization of POST requests to a RESTful API web service application. Each POST request will function through sending and receiving JSON files, ensuring a consistent format. To perform any POST requests to the database, the application had to include the following five step process: prepare the data, open a connection, send the data as a stream, assemble the response, verify the response code.

The current application had to have a method for preparing the data before submission was possible. This method consisted of three aspects; image handling, record handling, implicit order. For images to be submitted to the central database, they first had to be retrieved from the local database. Once the temporary image file for each log is allocated, the application had to convert it to a bitmap, and then, proceed to encode it as Base64 string. The pollinator records and user email will be assembled into a JSON file, with the proper field headings. However, the pollinator log would not contain the image location at this stage, because of the implicit order of submission due the structure of the central database. In the central database, images and pollinator records are stored independently, to reduce pollinator record query times. When an image is submitted it is stored on the central server in a folder for that particular user. Upon proper storage, the submission response code will contain the image path on the server. The image path is then attached to the pollinator record JSON, which will be inserted into the pollinator record table, on the server, for future data analysis. Having the submission process occur in separate steps can take the user a few seconds for a single record and significantly more for multiple records, however, simultaneously resulting in decreased server query times.

3.4.5 Viewing Submitted Logs

The previous application lacked the ability for users to access their submission history. The current application will allow users to access and inspect every log that has been submitted to the central database, that matches the user's email address. This feature became a possibility for inclusion after the introduction of two separate novel features, aforementioned as User Login

and Submission to the Central Database. Once these features were implemented, each submitted log will contain the user's login information and will be stored in the central database. Therefore, implementation of the ability to view submitted logs, required access to the central database and retrieval of all of the user's logs. This was accomplished via a GET request to the RESTful API web service application (REST APIs, 2008). A GET request functions by opening a connection to a specific URL in the web service and waiting for a response in a JSON format (REST APIs, 2008). However, to open this internet connection and retrieve the responses can take a significant amount of time. Simultaneously, the information requested from the database is necessary to create the screen the user is attempting to view. . Therefore, it became pertinent to retrieve this information on the previous screen, when the user selects to view submitted logs, enabling a smooth transition between screens. However, each log that is retrieved can also be inspected individually. Therefore, whenever a log is selected to be viewed individually, the related image is retrieved via a GET request to the central database (REST APIs, 2008). This saves the mobile device the work of loading all the images at once and prevents any memory problems.

3.5 Conclusion

This section has detailed the rationale and goals behind all of the updates being made to the application. The following section, Results, will describe the work that was completed and its functionality in detail.

4. Results

This section focuses on outlining the results of this project's work towards improving the previous project. Improvement of this project was focused around three main goals outlined as follows: improving functionality and accuracy of bee identification and information collection process, implementing functionality for transmitting data from local storage to central database, and improving user experience and application aesthetics. To accomplish these goals during development, it was important to focus on user feedback for updates to be made and on how to improve the work done throughout this project.

4.1 General Application Development Improvement

Our general development strategy focused around three main priorities: user experience, modularity, and stable functionality across multiple device manufacturers. With the intent of improving user experience, this project followed user feedback, and resolved any underlying issues within the application. This work meant removing portions of the application that would crash. This work also entailed reducing time spent identifying a pollinator, which was accomplished by reducing the number of screens in the identification process. To improve modularity and facilitate future development, many functions were handled and utilized consistently throughout the application, such as the process for requesting information from the database, and the method for loading new pollinator images. Lastly, the application had to be functional across device manufacturers. This was completed through utilizing user feedback, updating screens to have variable sizing, and ensuring the utilization of device defaults, such as the default image gallery when importing images. The source code of the new version of application was stored in BeecologyAndroidApp repository in GitLab. The development process and feature release of the application is stored in Gavin_Jacob_MQP branch which was a branch forked from the master branch that was developed by previous MQP.

4.2 The brand new icon of the Android application

Based on the feedback of the user experience with previous version of the application, the icon of the previous application had to be changed. User feedback indicated the previous icon, lacked any floral inspiration, disregarding the integral portion of pollinator research that is flower identification. Therefore, the current application has a new icon. The new icon incorporates three distinct yet important elements. First, the background of the new icon is inspired by an Aster, commonly referred to as the "Michaelmas Daisy". Aster is also recognized as a food source for various pollinators. Secondly, placed on top of the Aster, there is a cartoon style bumblebee. This cartoon bumblebee is displayed in flight, to instill a sense of motion to the icon. Lastly, the title of the application, "Bee-Cology" was reformatted and emboldened. This update will reduce difficulty users reported when attempting to find the application. After being installed on a user's phone the icon will appear as shown in Figure 3.

Figure 3: The current (left) and previous (right) icons. The previous icon is taken from (Olivia, 2017)

4.3 Logging a bee

This procedure was heavily updated during the course of this project. As a result, many new features and improvements were added to the procedure for users to log a bee. From these improvements, the new application will hope to provide a more efficient, accurate, and useful dataset, than could be achieved previously. All of the aforementioned features are outlined further below.

4.3.1 Select Capture

After clicking the “LOG A BEE!” button on the main screen of the application, users are being directed to a capture method selection screen. Compared to the old version of the application, the current application gives users more options to log a bee, therefore, this screen was reformatted. The current screen allows users to select from four capture methods. For layperson users, the application provides three capture options, “TAKE PICTURE”, “TAKE VIDEO”, or “IMPORT PICTURE”. For expert users, the application provides a method for logging a pollinator without an image, labelled “LOG AS EXPERT”. If users have previously enabled location permissions, then when users click on “TAKE PICTURE”, “TAKE VIDEO” or “LOG AS EXPERT” button, the application will automatically record the current location and timestamp. If the application lacks permission to access sensors or cannot get the current GPS information, the user receives a pop up saying “No Location Found!”. Users who view this message are encouraged to select “IMPORT PICTURE”; otherwise, the location field of the log will be shown as “No Location Found, Sorry”.

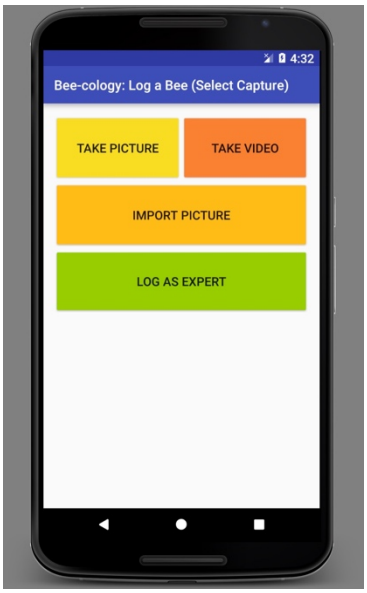


Figure 4: The select capture method screen.

4.3.2 Take Picture

Proceeding the Select Capture screen, if a user clicked the “TAKE PICTURE” button, the application will open the mobile device’s camera. This feature of the current application is not perceivably different from the predecessor.

4.3.3 Take Video

After the Select Capture screen, if user clicked the “TAKE VIDEO” button, the application opens the mobile device’s camera. After a video is captured, the user will be directed to the video editing screen. This screen will automatically loop-play a slowed down version of the video. On this screen the user has the ability to pause the video and to select an individual frame for capture. Following user feedback, this screen also features a scrolling seekbar under the video, which can be used to skip forward or backward in the video player.

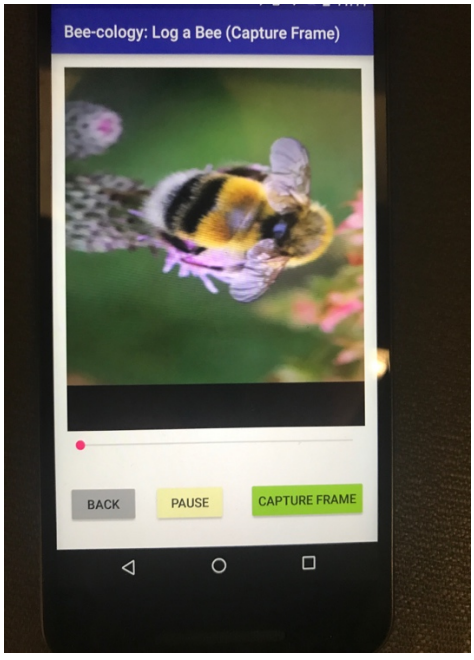


Figure 5: The video frame capture screen

4.3.4 Import Picture

Following user feedback, one of the new features that we added for the new application is to allow users to import pictures. If the user has selected the import picture method, the application will open the mobile device's photo gallery. This feature allows users to log any locally stored image. It is useful when user has multiple pollinators in a condensed region, in this case it is advantageous to take pictures first and create the log at a later time. Following selecting an image the user is directed to the Date/Location selector screen.

4.3.5 Date/Location Selector

Another feature included in the new application is the ability to select the date and location of which the log was recorded. This interface allows users to manually input where and when the log was recorded. The date can be selected via a drop down menu, which starts at the current date retrieved from the mobile device. The location can be selected through two separate methods, a drop down menu or a Google place picker. If the user selects to utilize the Google place picker, denoted by the blue "+" button, then the application will open a map, with which the user can pinpoint where the image was taken. When a location is selected via this method, it is saved locally on the mobile device and is included at the top of the drop down menu for ease of future location selection.

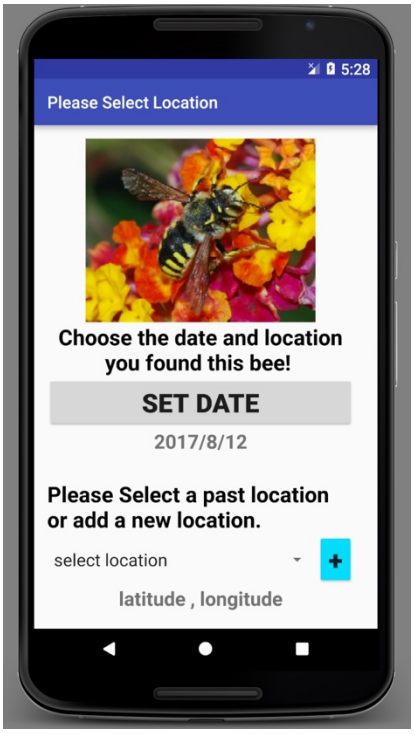


Figure 6: The date and location selection screen.

4.3.6 Choosing Guided or Manual Identification

After a user has decided upon and utilized an image capture method, they are prompted with two identification processes, Guided or Manual. This screen shows the user the captured image, date, and location for confirmation prior to proceeding. This screen has no perceivable updates compared to the previous version of the application, excluding the addition of date and location being displayed.

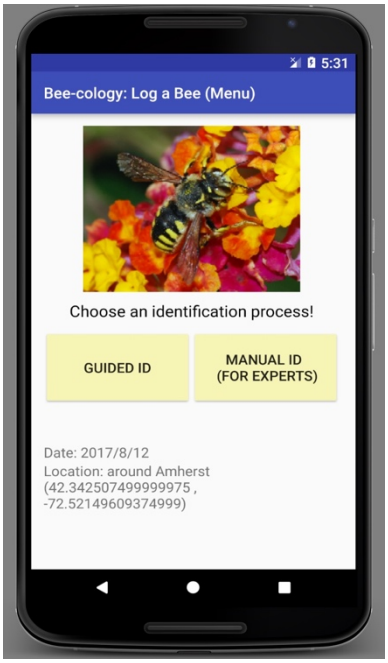


Figure 7: The select identification process screen

4.3.7 Manual (Expert) Identification

In the newest version of the application, there are two functions that allow users to identify a pollinator manually. The manual identification method is targeted for expert users who can accurately identify a pollinator species without guidance or visual aids. Following recommendations from user feedback, there are now two ways for an Expert to log a bee in this manner. Users can access the manual identification method from the select capture method screen, as well as the choosing guided or manual identification screen. In the previous application an expert user could only access the manual identification screen through the choosing guided or manual identification screen. This meant every expert log had to contain an image, which the expert would not need for assistance in identification, therefore, wasting the user's time. In this application, the expert can select manual identification from the select capture screen directly, without any related image, following this the user is prompted with the Date/Location selector screen prior to identification. Regardless of how a user selects the manual identification process, the application will eventually bring them to the following screen displayed as Figure 8, below. The manual identification screen contains two drop down menus, one for the species of

pollinator and another for gender. The drop down menus are an inclusive list of pollinators in the application and their relative gender variants. The bottom of this screen features a “SELECT FLOWER” button which will direct the application to the flower identification screen.

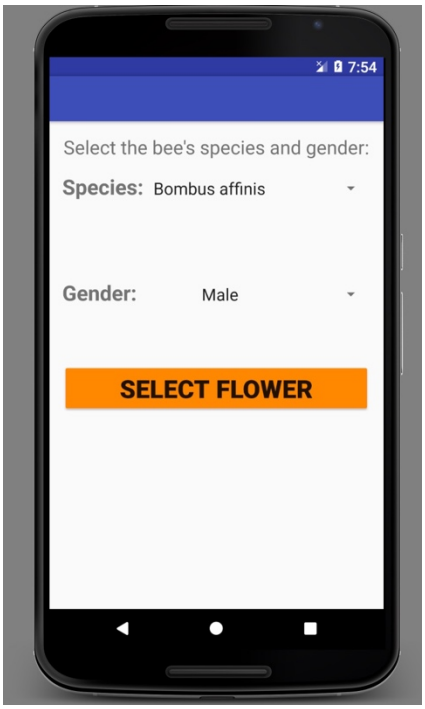


Figure 8: The expert ID screen

4.3.8 Guided Identification

The new application features a new process for the identification of pollinators. According to user feedback, the previous method was heavily time consuming and had no tolerance for mistakes. In the current version, users spend less time identifying and less effort on fixing any mistakes. The current application, has the following process for guided identification. First, the user is directed to a screen, Figure 9 below, with the options of: “Mostly Black”, “Half Yellow/Half Black”, “Mostly Non-Black”.

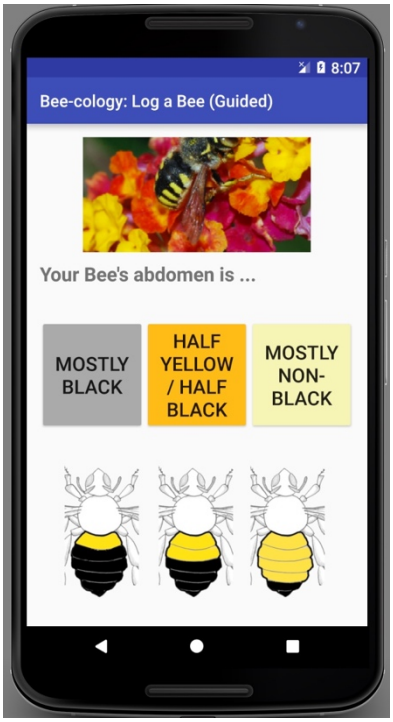


Figure 9: First screen of the guided ID process

Based upon the user's selection of basic abdomen coloration, the application loads multiple options for abdomen, thorax, and head coloration patterns for the user to select from. Each of these pollinator features is selected on independent screens displayed below, Figure 10. These screen all follow a similar layout in which the user is displayed with their image and a cartoon image of the pollinator, with unselected body regions greyed out. The user can switch between options for coloration patterns through the "arrow" buttons on each side of the body part name.

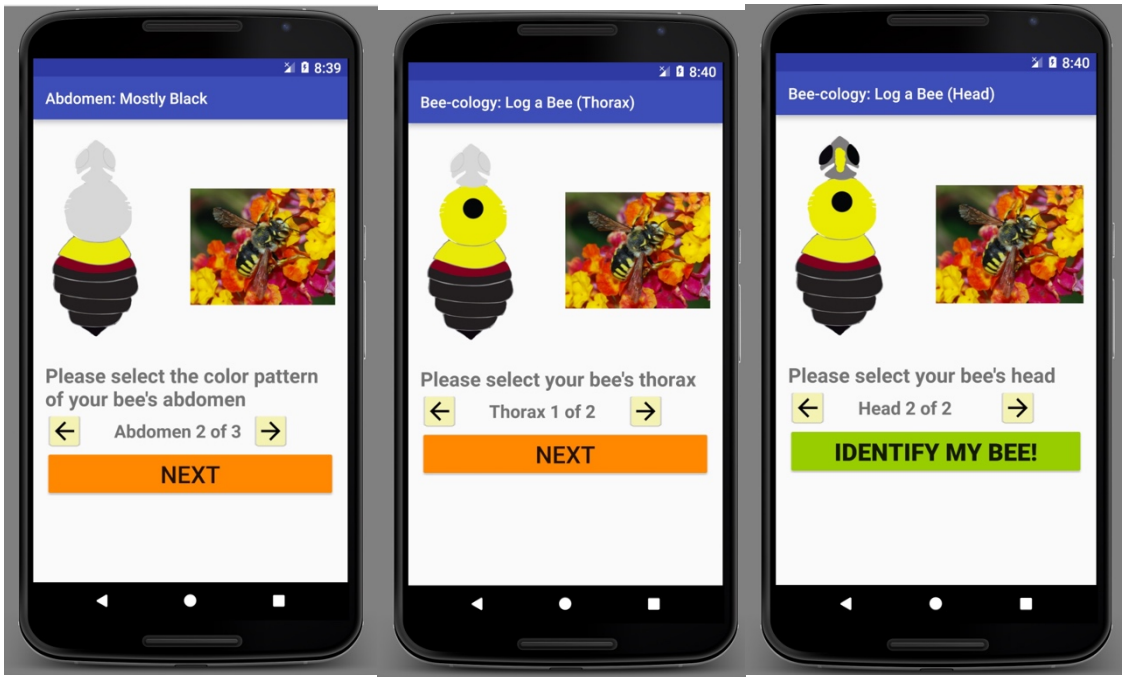


Figure 10: The new guided identification process. The screens allow the user to select from various coloration patterns for each body part, abdomen, thorax, and head, in that order.

Once all three factors of these regions are selected, the application gives the user feedback about the possible pollinator selected. If the user's selection of abdomen, thorax, and head could be accurately matched a pollinator in the local database, the user is directed to a confirmation screen show below on the left in Figure 11. If based on the user selected features, a pollinator can not be identified, or the pollinator selected does not appear accurate, then the user is directed to the error screen shown on the right in Figure11. On the first appearance of the error screen a user is prompted to attempt identification one more time. However, if after the second attempt at identification, the information still does not match with user's expectations or the pollinator can not be identified, the user can record the pollinator as an unknown species.

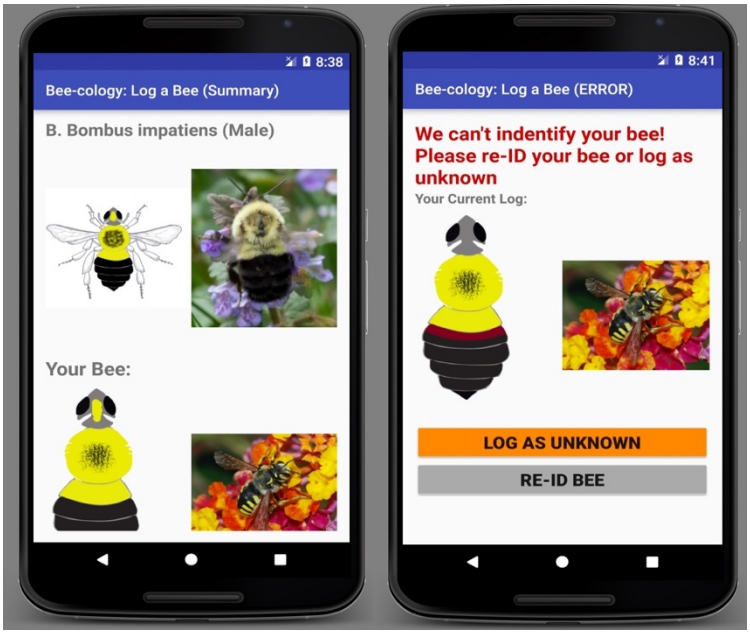


Figure 11: Results of the ID process screens. Based on previous user choices, the user is given these two screens where they must either confirm a match, left, or restart the identification process, right.

4.3.9 Flower Identification

The new version of the application maintains the similar method of flower identification compared to the previous version. However, it now includes a text box for user's to manually input the flower name/species and a drop down menu for selecting the pollinator's behavior, collecting pollen or nectar. The features hopes to collect novel information about the pollinators dietary preferences.

4.4 My Logs

The final features of the app revolve around the submission of the collected data to a central database. To properly accomplish this task, we have broken it down into a few integral parts: collection of user data, viewing current logs in the local database, inspecting individual logs, editing the logs, viewing previously submitted logs.

4.4.1 Collection of user data

The previous version of the application had no method of user identification, and therefore a reliability issue with the logs in the central database. As a result, the first time a user selects the MyLogs feature from the home page, the application prompts them to the enter their email address in two separate text fields. The application will only open this prompt for the user on the first use after installation. Any email the user inputs is validated, in the event the email is invalid, a pop-up message will inform them to try again.

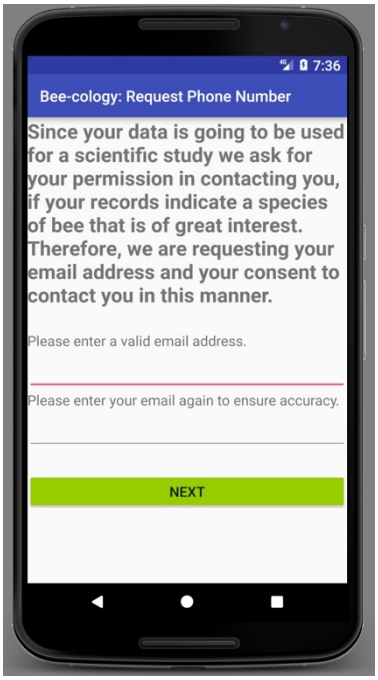


Figure 12: User login screen

4.4.2 Viewing current logs “My Logs” screen

The previous application contained a method for users to view the logs stored in their local database. This screen enabled users to scroll through their logs and select to review them individually. This screen does not perceivably differ from the previous version, excluding the addition of two new buttons labelled, “SUBMITTED LOGS” and “SUBMIT OR EDIT LOGS”. From this screen the user can select to submit/edit their current logs or they can view previously submitted logs. However, both of these options require an internet connection. Therefore, if the user attempts to access these features with impeded internet access, this screen will display various alert messages depending on internet status. If the mobile device has a data connection, that is not

Wi-Fi, the screen will prompt them to utilize Wi-Fi to ensure a stable connection, but access won't be impeded.

Figure 13: The my logs screen

4.4.3 Inspecting logs

The previous application allowed users to inspect each log individually via the inspect log screen. This feature had enabled the user to view all the details of a log. This screen has had only minor alterations, limited to the addition of new data fields being displayed. However, this screen display a log held in the local database or previously submitted logs from the central database, depending on which screen accesses it.

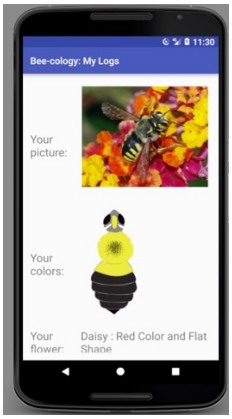


Figure 14: The inspect log screen

4.4.4 Edit Logs Screen

The previous version of the application did not feature a way to submit or delete logs stored on their mobile device. The current application, allows these features through the edit logs screen. This screen has a similar layout with my logs screen, allowing the user to scroll through logs and inspect local logs individually or activate a check box next to each. On this screen it is recommended that a user review their logs for accuracy. If a log is inaccurate the user can check it off and select to delete the checked logs, or delete all of the logs at once. If a log is deleted, then all related information and the image associated is deleted. After removing unreliable logs, a user selects to submit the logs to the central database, individually or all at once. When a user attempts to submit they will be prompted to wait during the submission process. After submitting the pollinator record as well as the related image, this screen will create a prompt thanking them for their submission. Upon a successful submission the log will be deleted to prevent multiple submissions of the same record.

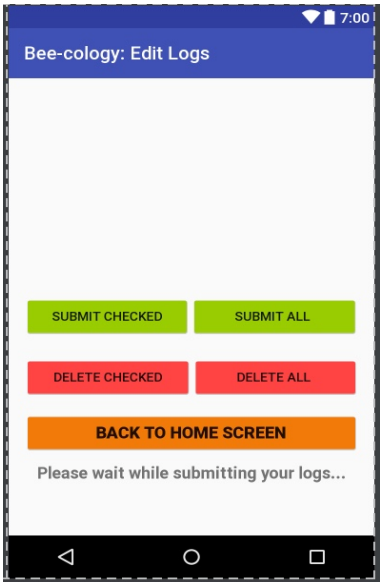


Figure 15: The edit logs screen

4.4.5 Viewing submitted Logs

With the addition of the ability for a user to submit logs, it became necessary and possible for the current version of the application to include a screen for viewing the logs the user previously submitted. This screen is accessed through my logs screen, and follows a similar aesthetic. The only logs displayed are ones submitted with the current user's email address attached. They will will be listed off by date and pollinator species, and each can be inspected individually.

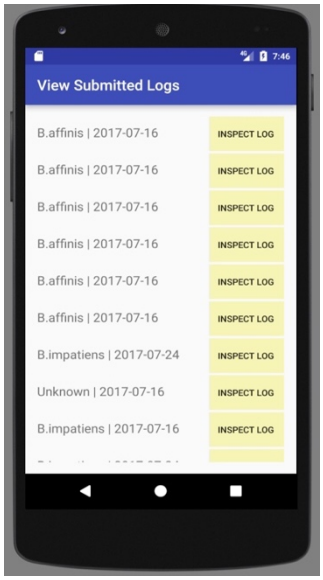


Figure 16: The view submitted logs screen

5. Discussion

The overarching goals of the Bee-cology project focus on gathering large quantities of data about pollinators for data analysis and for ecological modeling. Our MQP worked on improving and creating a functional mobile application enabling data collection to be crowdsourced, representing the first major step towards completing the overall goals of this project. This MQP was successful in improving the functionality and user experience of the previous application, to which end the application can now be distributed and utilized by the public. Simultaneously, updates to the identification process help to ensure that a higher rate of submitted data is accurate. As a result, public usage is already growing and resulting in a substantial increase in collected data about pollinator biodiversity. The previous application enabled users to identify pollinators and record data to their mobile device's local storage. This provided a solid foundation for this project to expand upon with respect to the following goals: improving functionality and accuracy of bee identification and information collection process, implement functionality for transmitting data from local storage to central database, improving user experience and application aesthetics. The current state of the application has fulfilled the majority of these goals underlying requirements and enabled the application to be released to the general public, therefore this project has reached its successful conclusion.

While this project reached a successful conclusion, there is still work to be completed to further the usage and effectiveness of this application. A few features which need improvement on the backend development side are listed as follows: memory usage, location selection, central database submission, and modularity of the application. Some of the screens in the application can utilize a large portion of the RAM in the mobile device, occasionally causing the application to crash. This problem was solved temporarily by shifting when the application does certain operations, such as loading previously submitted logs. This problem could be solved permanently by allowing the user to select the number of logs to be loaded and viewed, and subsequently informing them if their request is too large for the mobile device. The current method for location selection can be unstable depending on the user's internet connection, which can result in logs lacking location information upon submission. This method will need to be updated to function independently of internet or allow location to be added to the log later. The current method for

submitting logs to the central database is extremely time consuming and has stability issues depending on internet connection. To solve this issue, submission could be done in a single POST request, rather than multiple per log, decreasing time and connection required to submit multiple logs. Lastly, the backend of the application can be greatly improved by increasing its modularity. Many functions are re-created and implemented differently depending on the screen, therefore, making updates to the application more difficult than re-coding certain aspects. To improve this aspect, many functions should be imported as helper functions rather than re-written multiple times. For example, this improvement could be applied to the "setPic" function, which is used in each screen with an image, but with different sizing and resolution.

This application could also be improved upon by increasing the quality of information gathered. To accomplish this, future projects could work on improving the flower identification method and allowing logs to be edited prior to submission. The current method for flower identification does not actually identify the flower, but instead has the user input basic traits and possibly a flower name. However, this information is incomplete and does not reliably allow a flower to be identified. Therefore, this method should be re-designed to obtain similar quality of results as the pollinator identification. Secondly, while a user can view logs prior to central database submission, users are unable to edit them. If a user is allowed to edit their logs, then hopefully, inaccurate logs can be corrected rather than deleted before submission, increasing the quality and quantity of information gathered.

One of the significant goals of this MQP was to improve the application aesthetics and therefore the user experience. However, this goal could not be reached as it was more pertinent to improve functionality. To improve aesthetics requires a complete overhaul of almost every screen. The current application features simple colors and blocky design which isn't very attractive to the user. Future iterations of this application should have re-designs of each screen to resemble most modern mobile applications.

This project improved upon the previous application by enabling enough functional features for the application to be utilized by the public. However, work on this application is not over yet and as such new project teams have already formed. The next IQP team is working on increasing the quantity of data collected through the creation of a website, which mimics the applications functionality. The next MQP team hopes to implement a new aesthetic and to increase

overall stability of the application. If future work enables the application to be completely stable and have an interesting and appealing design, then hopefully, the application will become available to all android users through the Google Play store.

References:

About SQLite. (2017). Retrieved August 27, 2017, from *https://www.sqlite.org/about.html*

August, T., & Pocock, M. (2017). Pros and Cons of Citizen Science. Speech. Retrieved from: http://www.snh.gov.uk/docs/A1470291.pdf

Bar, A. R., & Maheswaran, M. (2014). Integrity Management. In Confidentiality and integrity in crowdsourcing systems(pp. 39-57). Cham: Springer.

Goulson, D. (2012). Bumblebees behaviour, ecology, and conservation. Oxford: Oxford Univ. Press. Retrieved from: *http://ebookcentral.proquest.com.ezproxy.wpi.edu/lib/wpi/detail.action?docID=472276*

Jackson E. Oliva. "Citizen Science Bee Ecology". Major Qualifying Project. Worcester Polytechnic Institute. April 27, 2017.

Kenedi E Heather, Rachel Lee Murphy, and Devin T Stevens. "Bee-cology: Educating the New England Area on the Importance of Pollinator Diversity". Interactive Qualifying Project. Worcester Polytechnic Institute. March 2017.

Meet Android Studio. (2017, August 03). Retrieved August 27, 2017, from *https://developer.android.com/studio/intro/index.html*

PlaybackParams. (2017, July 24). Retrieved August 20, 2017, from *https://developer.android.com/reference/android/media/PlaybackParams.html*

Providing Proper Back Navigation. (2016, September 01). Retrieved August 20, 2017, from *https://developer.android.com/training/implementing-navigation/temporal.html*

REST APIs . (2008, October 20). Retrieved September 04, 2017, from *http://roy.gbiv.com/untangled/2008/rest-apis-must-be-hypertext-driven*

Status of pollinators in North America. (2007). Washington, D.C.: The National Academies Press.

Swagger Specification. (2017). Retrieved August 27, 2017, from *https://swagger.io/specification/*

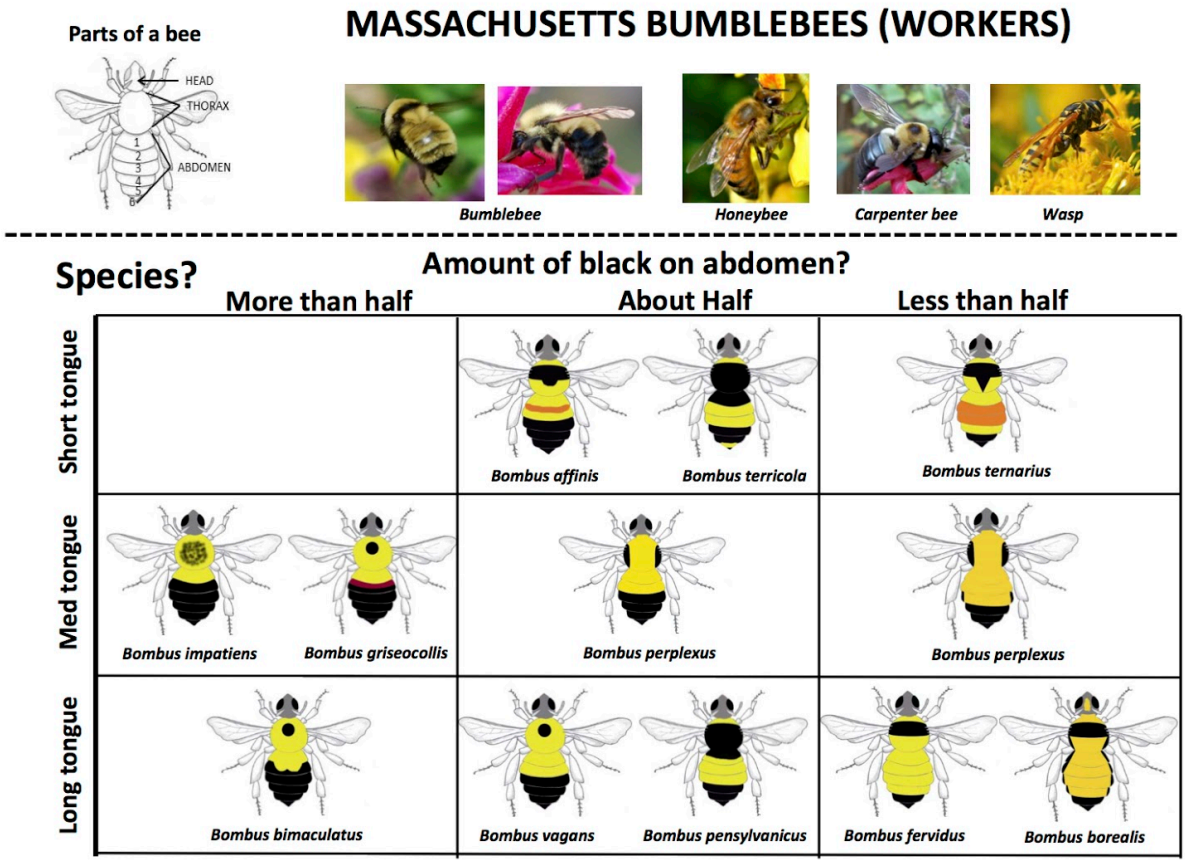
What is citizen science? (2017). Retrieved September 04, 2017, from *https://scistarter.com/page/Citizen%20Science.html*

Williams, P. H., Richardson, L. L., & Thorp, R. W. (2014). Bumble Bees of North America: An Identification Guide. Princeton University Press. Retrieved from: *http://ebookcentral.proquest.com.ezproxy.wpi.edu/lib/wpi/reader.action?docID=1604276&ppg=18*

Appendix A:

Cartoon Color pattern for Bumblebee Species:

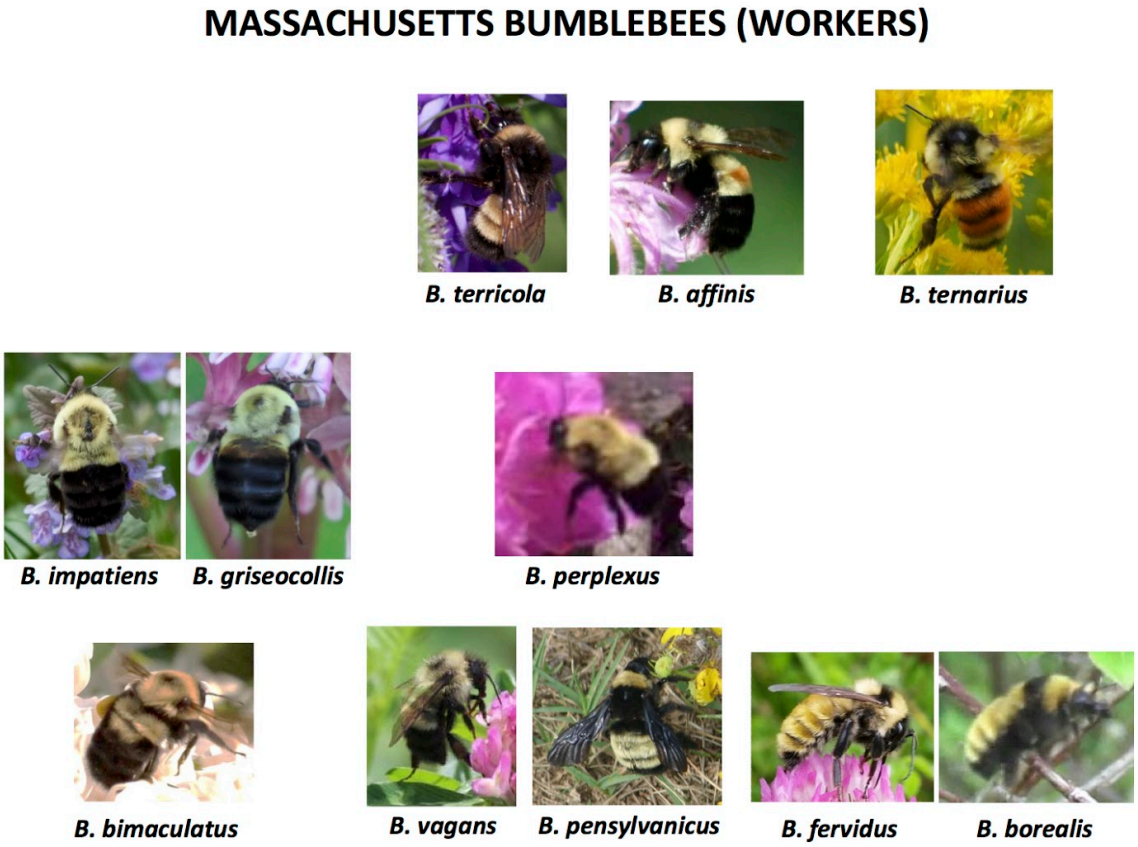
Provided by Professor Robert Gegear's Lab



Appendix B:

Bumblebee Field Images in Application:

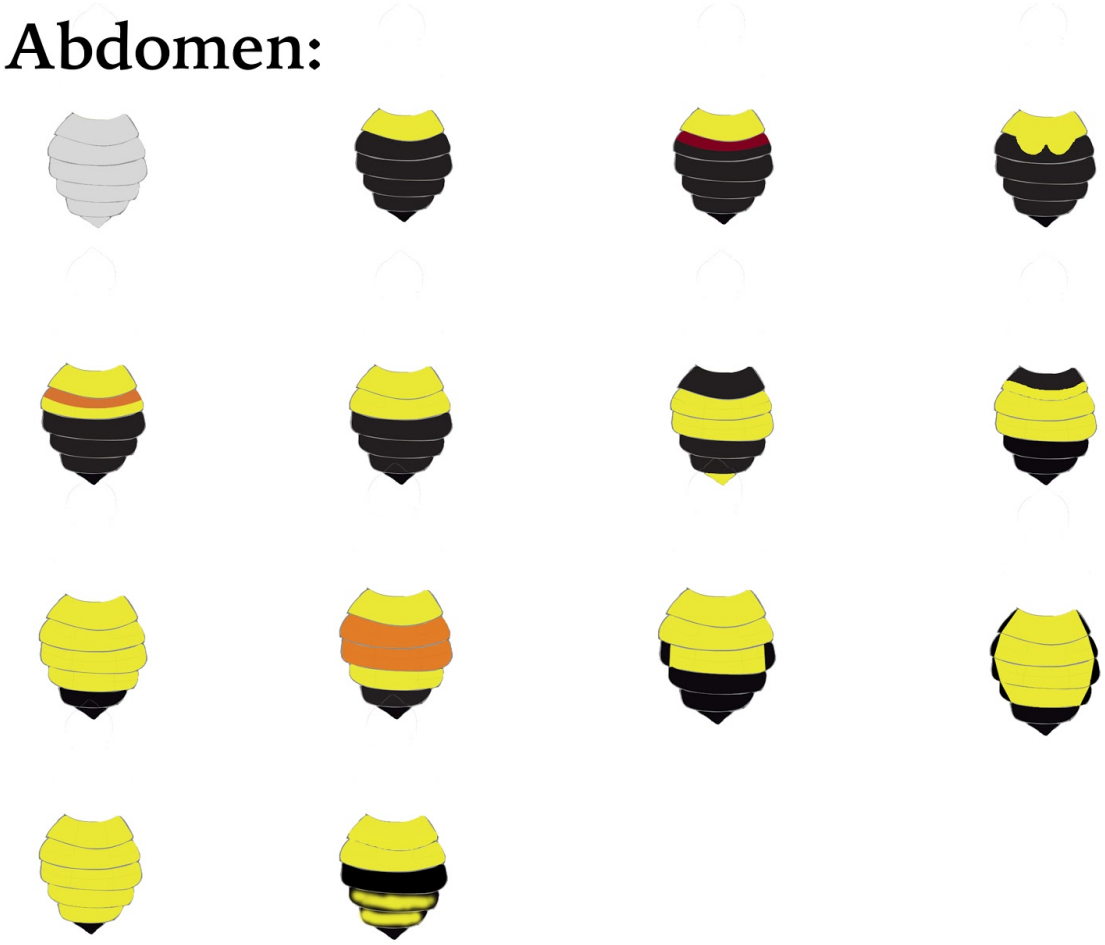
(Provided by Professor Robert Gegear's Lab)



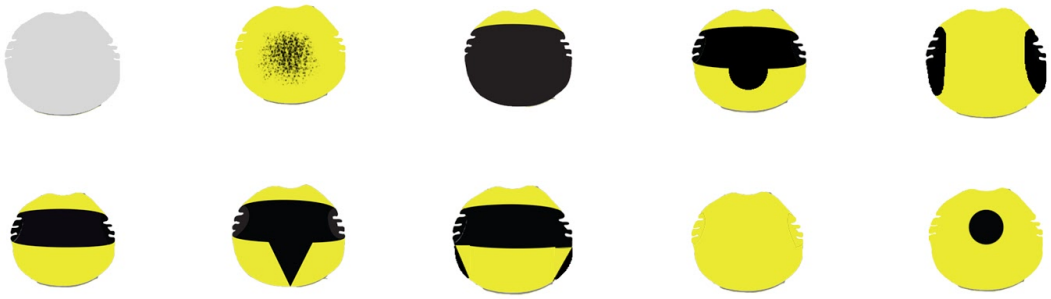
Appendix C:

Abdomen, Thorax and Head Options:

(Modified from (Heather, Murphy, and Stevens, 2017))



Thorax:



Head:

Appendix D:

Bumblebee Identification Tree:

(Taken from (Heather, Murphy, and Stevens, 2017))