How to Use this Template

1. Make a copy [File → Make a copy...]
2. Rename this file: "**Capstone_Stage1**"
3. Replace the text in green

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"

GitHub Username: rodrigoshariff

Birder Journey

Description

Birder Journey allows outdoor enthusiasts to learn, identify and keep track of the bird species that they see in an easy and organized way. With the search functionality, the user can access a list of species and then access the detail information of each species where he/she will be able to see an image and relevant information and then, if appropriate, log the observation on the field. The app will store the species, optional note, date and location of each observation

and will provide the ability to create a summary of distinct counts and lists by specific day, current week, month, year or life.

Intended User

The intended user ranges from beginner bird watchers to experienced birders in North America looking for a tool to view images, read bird information and log each observation for future reference.

Features

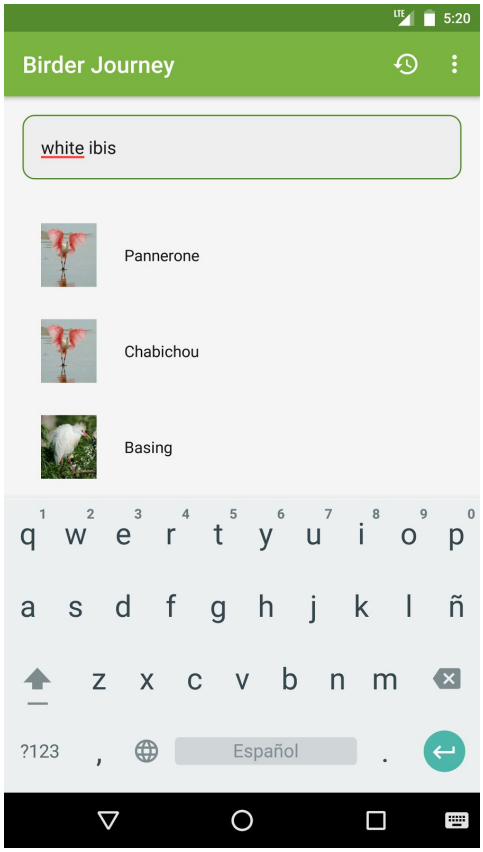
The main features of the app are:

- Includes searchable information for all bird species in the November 2015 ABA (American Birding Association). Upgradable without affecting users via Google Cloud Endpoint.
- Most species are illustrated with high resolution images and current IUCN Red list conservation status.
- Each observation of a species of bird can be stored along with its location, date, time as well as a user note.
- Provides detailed daily summary of observations as well as general daily, weekly, monthly, yearly or life long bird species lists with distinct counts. General lists can be drilled down to individual observations.

User Interface Mocks

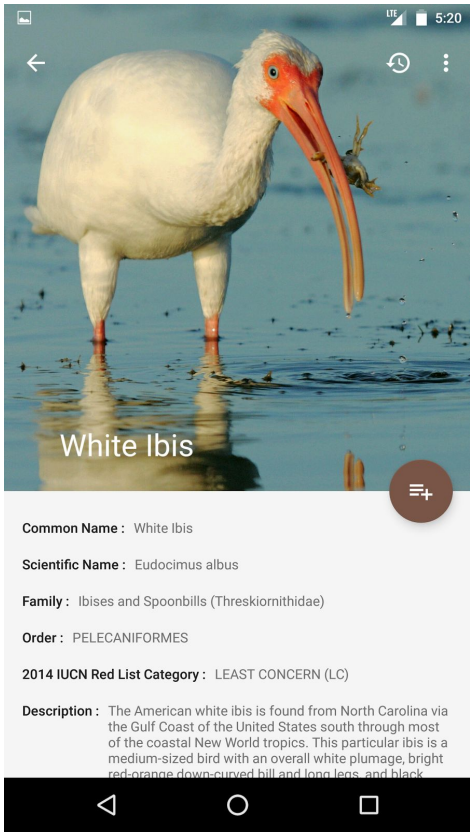
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

Screen 1



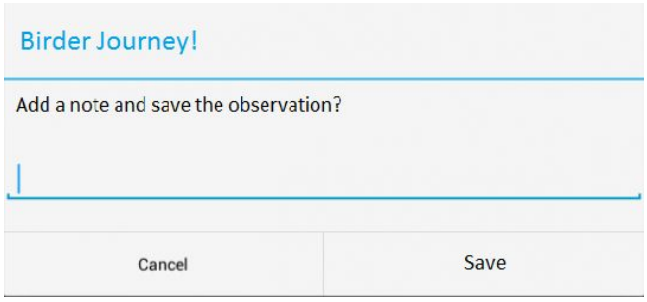
Entry screen allows the user to search for a species of bird by using the name. The results are displayed in a listview that will take the user to the detail page (Screen 2) when pressed.

Screen 2



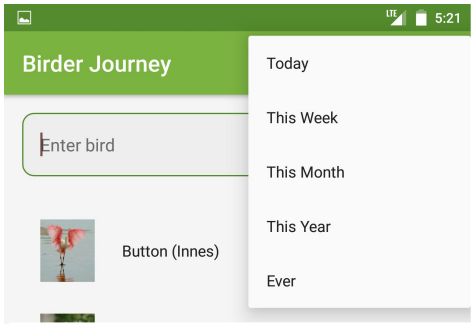
Detail page includes a high resolution image (most if not all images are of my own creation) of the species and information such as scientific name, family, order, etc. An alternate layout will be use for landscape orientation on phones.

Screen 3



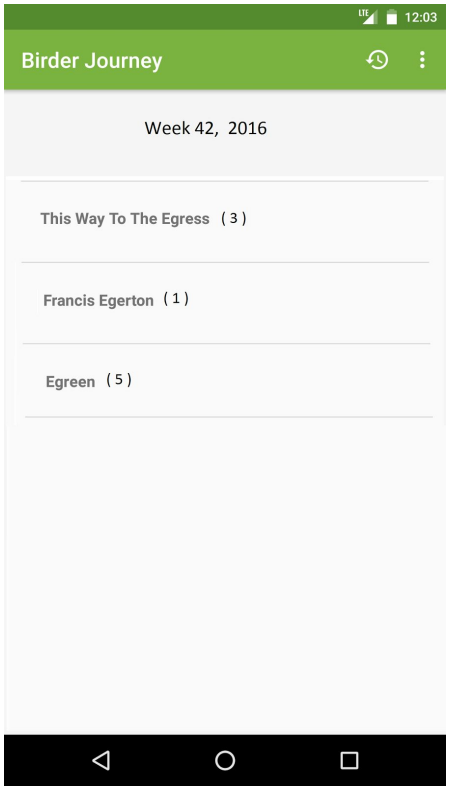
The FAB in Screen 2 will start an alert dialog (Screen 3) that will allow the user to input a note associated with the observation and save it permanently.

Screen 4



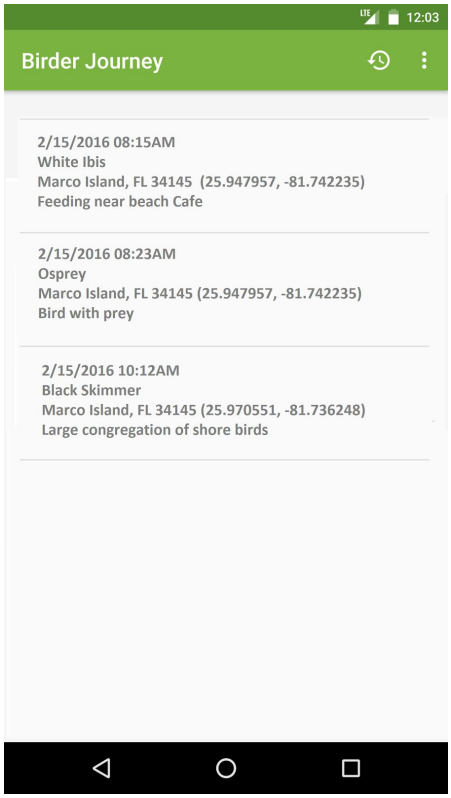
The toolbar of the app has an icon that will open a menu for the user to choose a timeframe for the summary report of his/her historical observations. This options will take the user to another screen with a listview displaying all distinct species observed with counts in the timeframe selected (Screen 5). An option to pick a specific date (date picker) for a detail report is possible as an option.

Screen 5



General report displaying distinct list of species with observation count for each item. Pressing an item here will take the user to the detail report of each individual observation (Screen 6).

Screen 6



Detail report displaying all the information associated with each observation including date/time, species of bird, nearest city (latitude, longitude) and user note. An application icon will be added to the toolbar that will navigate up when pressed.

Tablet support

To accommodate tablet users, a multiple fragments, multiple activities technique will be used for multi-pane screens that will manage the fragments illustrated above.

Add as many screens as you need to portray your app's UI flow.

Key Considerations

How will your app handle data persistence?

SQLite and a Content Provider will be built to persist user log/history. Loaders will be used to populate views.

Describe any corner cases in the UX.

The main activity/fragment will drive all navigation allowing the user to either view detail screen or report screens.

Describe any libraries you'll be using and share your reasoning for including them.

I will be using a Google Cloud Endpoint to store all bird information including image URL. This will service the search functionality.
I will be using Picasso to handle the loading and caching of bird images.
I will be using the location provider in Google Play Services to get location and current city for each saved observation. This information will be displayed on the detail report.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

Task 1: Project Setup

- Finalize all content (bird images, descriptions, etc) and
- Load to GC Endpoint to allow for querying/searching.
- Setup Content Provider and SQLite database to store user observation information including bird id, location, and user note.

Task 2: Implement UI for Each Activity and Fragment

- Create layout for main activity / fragment (Search)
- Create layout for detail fragment (image and bird information). Also handle landscape orientation.
- Create dialog fragment for entering note and saving observation.
- Create layout for general and detail report fragments.

- Create main - detail layout for tablet

Task 3: Your Next Task

- Implement search functionality to query GCE using AsyncTask or intent service. Populate listview with results.
- Start detail screen with selected species.
- Create SQL statements to insert new observation.
- When FAB in detail screen is pressed, use location provider in Google Play Services to get location and reverse geocode to get nearest city of the observation and insert this along with other information to the database.
- Create statements to query for general summary and detail summary.
- Use loaders to interact with content provider and populate listviews.

Task 4: Use fragment managers to load fragments in appropriate activity based on screen size.

Task 5: Set up Gradle for release flavor build and implement signing configuration.

Task 6: Error handling (Network, etc), input validation.

Task 7: Set up Accessibility (Content descriptions, RTL, etc)

Task 8: Implement overflow menu of the toolbar to include a feedback option as well as credits/references.

Task 9: Create Widget

Add as many tasks as you need to complete your app.

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"