**GitHub Username**: gbielanski

# What's this?

## Description

How many times you saw something and did not know what is this? Image there is no need to describe what you see but just make a photo and let your app to find information at Wikipedia. This is what **What's this?** do.

## Intended User

This application is for everyone.Using this application is as easy as taking a photo. You need to be able to read so it is not aimed at children under the age of 8.

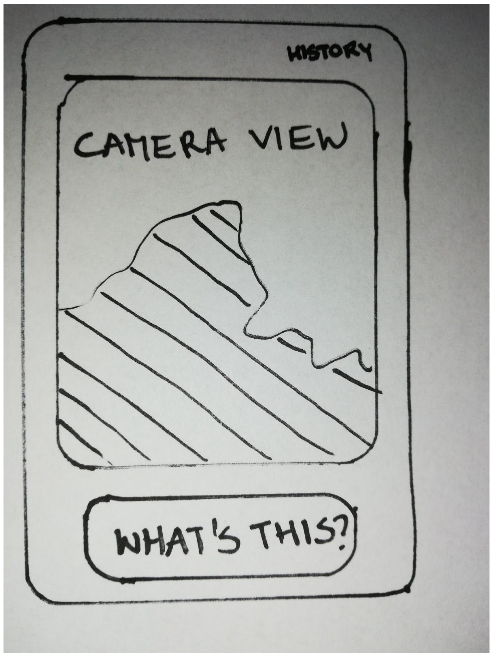
## Features

List the main features of your app. For example:

- Takes pictures
- Provides list of possible subjects to find
- Saves information about chosen subjects and taken photo

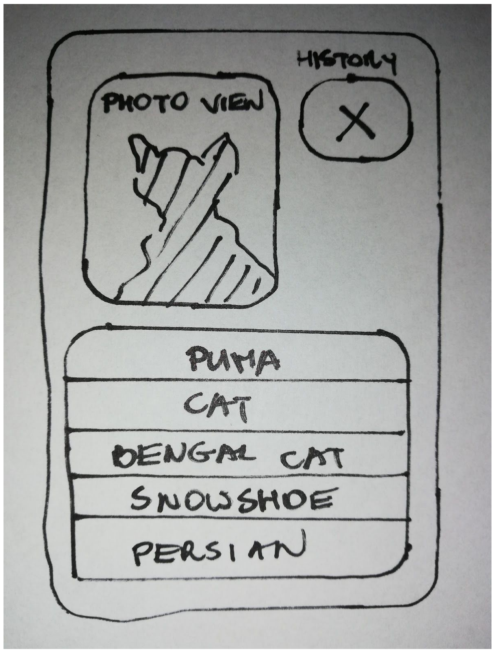
## User Interface Mocks

### Screen 1



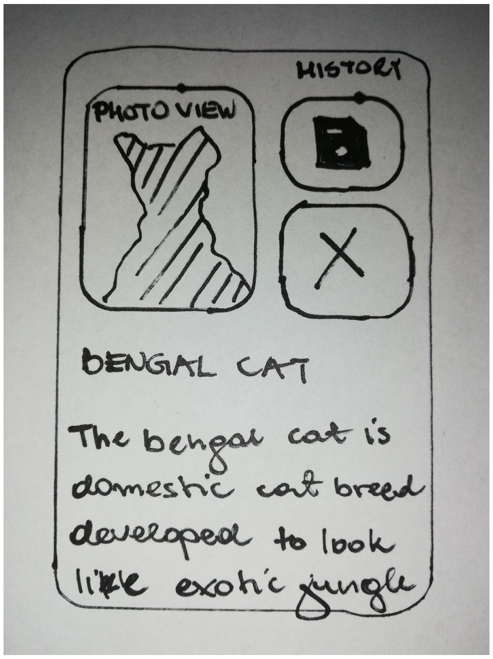
First screen shows camera preview. It has one button with text “What’s this?” which captures a picture. App menu “HISTORY” takes to the screen 4.

## Screen 2



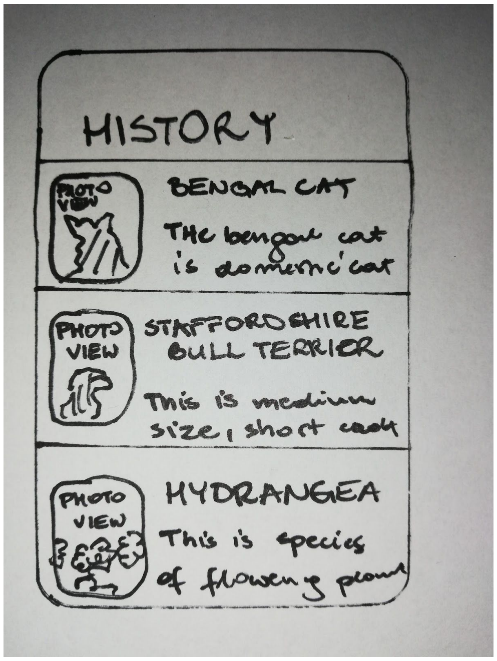
Screen contains picture captured at screen 1. List contains five most likely photo content. Pressing for one of list item takes to the screen 3. Button “X” takes back to the first screen. App menu “HISTORY” takes to the screen 4.

## Screen 3



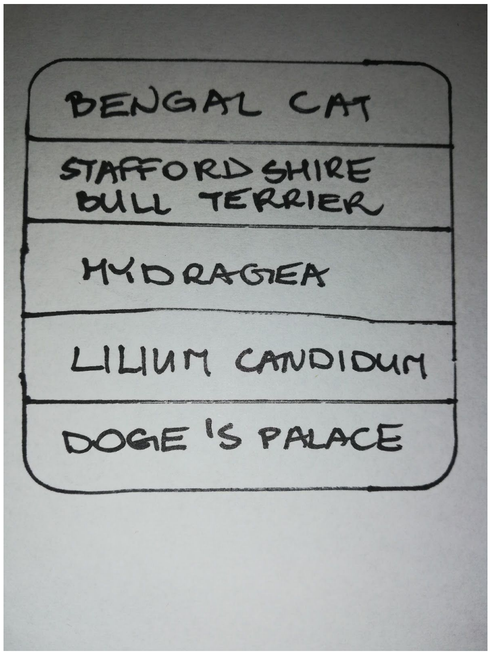
Screen contains picture captured at screen 1. Wiki information base on option taken from the list from screen 2. There are two buttons. First “Save” stores information and photo and those information will be visible in “HISTORY” screen. This button is not visible when started from screen 4. Second Button “X” takes back to screen 2. App menu “HISTORY” takes to the screen 4.

## Screen 4


HISTORY
PHOTO VIEW
BENGAL CAT
THe bengal cat is domestic cat
PHOTO VIEW
STAFFORDSHIRE BULL TERRIER
This is medium size, short coat
PHOTO VIEW
HYDRANGEA
This is species of flowering plant

Contains information stored using button at screen 3. This is RecyclerView. Pressing any item takes to the screen 3.

## Widget



Widget presents list of last a few saved subjects.

## Key Considerations

Application will be written in Java.
Android Studio 3.1.3
Gradle 4.4

### How will your app handle data persistence?

App will handle data using Room Persistence Library.
Use MVVM (LiveData, ViewModel and Room)

### Describe any edge or corner cases in the UX.

Photo will be material design shared element between screen 2 and 3.
Screen 3 will have scrollable description.
Button “X” takes back as back button.
Screen 4 will contains RecyclerView.
App using Material Design colors
App containing contentdescription to support accessibility
App Enable RTL
Strings are kept in strings.xml
App using styles.xml file to store repeating elements of layouts

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

Timber version 4.7.1
Picasso version 2.71828
Room Persistence Library 1.1.1

### Describe how you will implement Google Play Services or other external services.

Google Cloud Vision Api
Wikipedia Api

## Next Steps: Required Tasks

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

## Task 1: Project Setup

- Configure libraries
- Setup Google Cloud Project

## Task 2: Implement UI for Each Activity and Fragment

- Build UI for Screen 1
- Build UI for Screen 2
- Build UI for Screen 3
- Build UI for Screen 4

## Task 3: Photo capturing

- Camera preview
- Taking photo and saving

## Task 4: Google Cloud Vision Api

- Sending photo to google cloud
- Getting and parsing response

## Task 5: Screen 2 data

- Feeding ListView with data from Vision Api
- Reading and displaying saved photo
- Use AsyncTaskLoader to read data from VisionApi

## Task 6: Screen 3 data

- Taking information from Wikipedia Api
- Feeding screen 2 by received information
- Reading and displaying saved photo
- Use AsyncTaskLoader to read data from internet

## Task 6: History

- Saving information on screen 3
- Creating RecyclerView and Adapter for screen 4

- Feeding screen 4 by data from database
- Staring screen 3 with data from database without save button
- Use AsyncTaskLoader to read data from database

## Task 6: Widget

- Add Widget UI
- Create Widget Provider class
- Create Widget RemoteViewsService

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

---

**Submission Instructions**

- After you've completed all the sections, download this document as a PDF [ File → Download as PDF ]
    - Make sure the PDF is named "**Capstone_Stage1.pdf**"
- Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

- Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
- Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"