**Lone Wolf Coding Project Summary**

**

**Tyler Pearson, Jasmine Gutierrez, Mansi Chaubey**

**CS 440**

**at the**

**University of Illinois Chicago**

**September 2018**

**Table of Contents**

[*REMOVE OR REPLACE ALL TEXT IN RED ITALICS BEFORE SUBMITTING REPORT* 2](#_gjdgxs)

[*How to Use This Document* 2](#_30j0zll)

[List of Figures 6](#_1fob9te)

[List of Tables 7](#_3znysh7)

[I](#_2et92p0) Project Description 8

[1](#_tyjcwt) Project Overview 8

[2](#_3dy6vkm) Project Domain 8

[3](#_1t3h5sf) Relationship to Other Documents 8

[4](#_4d34og8) Naming Conventions and Definitions 8

[4a](#_2s8eyo1) Definitions of Key Terms 8

[4b](#_17dp8vu) UML and Other Notation Used in This Document 10

[4c](#_3rdcrjn) Data Dictionary for Any Included Models 10

[II](#_26in1rg) Project Deliverables 11

[5](#_lnxbz9) First Release 11

[6](#_35nkun2) Second Release 11

[7](#_1ksv4uv) Comparison with Original Project Design Document 11

[III](#_44sinio) Testing 12

[8](#_2jxsxqh) Items to be Tested 12

[9](#_z337ya) Test Specifications 12

[10](#_3j2qqm3) Test Results 13

[11](#_1y810tw) Regression Testing 13

[IV](#_4i7ojhp) Inspection 13

[12](#_1ci93xb) Items to be Inspected 14

[13](#_3whwml4) Inspection Procedures 14

[14](#_2bn6wsx) Inspection Results 14

[V](#_qsh70q) Recommendations and Conclusions 14

[VI](#_3as4poj) Project Issues 14

[15](#_1pxezwc) Open Issues 14

[16](#_49x2ik5) Waiting Room 15

[17](#_2p2csry) Ideas for Solutions 16

[18](#_147n2zr) Project Retrospective 16

[VII](#_23ckvvd) Glossary 17

[VIII](#_ihv636) References / Bibliography 17

[IX](#_32hioqz) Index 18

### List of Figures

[Figure 1 - Activity Diagram of Lone Wolf](#_1hmsyys) **Page 7**

[Figure 2, 3, 4 - Code Samples from Source Code](#_41mghml) **Page 20, 21**

### 

# Project Description

## Project Overview

Lone Wolf is an app that is designed for visitors that do not want to be in a large tour group with other people when visiting a national park, as well as for experienced visitors to national parks looking for new experiences at the parks by finding more challenging/unused paths in them.

## Project Domain

The domain of the project revolves around the mobile development of the project and how it was implemented, as well as how the project interacts with the various APIs and database products used in its development.

## Relationship to Other Documents

This document is essentially a report on how Group 13 of the Spring semester of 2021 implemented the Lone Wolf development document written by Group 6 of the Fall 2020 semester.

## Naming Conventions and Definitions

JUnit: The standard testing framework used in Java development and used specifically in this project

androidx.test: API used in tandem with JUnit with making and running the smaller tests

Espresso: API used in tandem with JUnit and androidx.test to make and run the larger tests on the product, especially with the UI elements in the project.

### Definitions of Key Terms

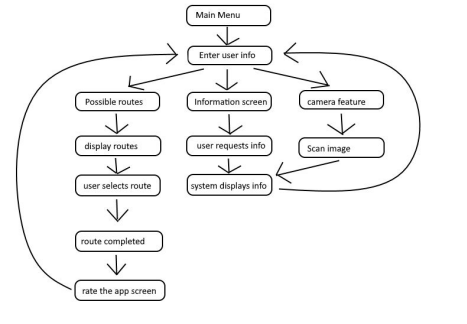
**Park:** used when referencing the National Parks

**Phone:** references smartphones rather than all phones together

**Interactive Map:** references a map that adjusts to a current user’s location. Increases the cost of the product due to having to implement it into the product with a license

### UML and Other Notation Used in This Document

This document follows the OMG UML standard as described by Fowler in [4]. The main one used is the Activity diagram as shown below:

****

### Data Dictionary for Any Included Models

Emergency Requests: Inside database as pushed in by client application, consists of android Location object with Latitude and Longitude defined and specified inside object.

Park Lookup Information: Name of Park, Latitude/Longitude, State(s) Park is Located in; Dynamically looked up and downloaded onto app from National Parks API

User Account: First Name, Last Name, User Email, Password; Stored in authenticator

Picture Upload: Uploaded photo; Stored in database storage

# Project Deliverables

The implemented functions of the project are as follows:

* Account Creation
* Account Recovery
* User Login
* User Logout
* Finding a Route in a Park
* The Encyclopedia
* Park Information Lookup
* Photo Uploading
* Emergency Requests

## First Release

The first release was on February 26th, and the functionalities included were the account creation, encyclopedia, and park route lookup.

## Second Release

The second release was on April 2nd, and all functionalities listed from the first release and all others mentioned in this section of the document were implemented by this time.

## Comparison with Original Project Design Document

The produced prototype did implement most of the features listed by the design team in the design document[5], but there were some features that were not implemented in the prototype.

The weather functionality listed in the document on page 3 was not implemented, as the development team felt that since there was a weather application natively installed on most every smartphone being used in the world so far, not to mention the other weather applications in existence, that implementing another version of this would be a waste of time and resources, as this would not give us any edge over the competition.

The interactive map as referenced first on page 2 was only somewhat implemented, as there was no need to have any weather simulated on the map on account of there being no weather functionality in the project itself. This was once again seen as being a drain on resources and time that we could not reasonably afford to waste.

The project was originally designed to be used on both iOS and Android devices, but the prototype was only made for Android. At the time, only one member of the development team had a Apple computer that would be capable of developing for iOS, and the team was under the impression that this was a necessity for iOS implementation, and the team did not know of cross-platform development technologies like Flutter and Xamarin. Had the team known about these technologies, the team would have implemented the project in both systems as specified by the design team.

# Testing

## Items to be Tested

* ID# 100 - Name: Monitor User Location
* ID# 101 - Name: User Login
* ID# 102 - Name: User Registration
* ID# 103 - Name: Retrieve Route
* ID# 104 - Name: Park Encyclopedia
* ID# 105 - Name: Upload Images
* ID# 106 - Name: Display Images
* ID# 107 - Name: Ask For Help
* ID# 108 - Name: Speed Test
* ID# 200 - Name: Developer Access Only
* ID# 201 - Name: Limited Visitor Access
* ID# 300 - Name: Credible Data Provided
* ID# 301 - Name: Protect User Info

## Test Specifications

**ID# 100 - Name: Monitor User Location**

**Description:** The app should be able to monitor user location at all times while using the app.

**Items covered by this test:** User Location

**Requirements addressed by this test:** Functional Requirements

**Environmental needs:** The app should be running on a smartphone and the user should have allowed location sharing with the app.

**Intercase Dependencies:** NA

**Test Procedures:** Once the app is running on a smartphone and is connected to the internet, the user location should be accessible by the app at all times. This can be tested by using the app at different locations and checking if the location changes in real time.

**Input Specification:** User should allow the app to monitor location.

**Output Specifications:** If the user allows the app to monitor location while using the app, then the location on the app should change in real time according to the location of the user.

**Pass/Fail Criteria:** If the app can show changes in user location in real time, then this test passes. Otherwise, it fails.

**ID# 101 - Name: User Login**

**Description:** Once a user has registered on the app, the app database should have their info, and should be able to authenticate future logins.

**Items covered by this test:** User login and user registration.

**Requirements addressed by this test:** Functional requirements.

**Environmental needs:** Firebase has been used as the app database. User information should be added to the firebase database.

**Intercase Dependencies: ID#102 -** User Registration Test.

**Test Procedures:** Once a user is registered, check the firebase console to find the user info in the User Information database.

**Input Specification:** The email address and password should be registered and should match each other according to the information stored in the database.

**Output Specifications:** If the information is correct, the user should be able to sign into their account. If the userid/password is incorrect, the user should be informed and prompted to use the “forgot password” function or to register as a new user.

**Pass/Fail Criteria:** The test passes if once a user is registered and enters the correct id and password, they are able to access their account.

**ID# 102 - Name: User Registration**

**Description:** If the user does not have an account with the app already, they have the option to register as a new user by entering their information.

**Items covered by this test:** User Registration.

**Requirements addressed by this test:** Functional requirements.

**Environmental needs:** Firebase has been used as the app database. User information should be added to the firebase database.

**Intercase Dependencies:** NA

**Test Procedures:** If a user successfully registers, their information should be added to the user information database and can be accessed on the firebase console by the developers/app.

**Input Specification:** The userid should be a valid email address and the password should meet the set requirements.

**Output Specifications:** After registering successfully, the user should be redirected to the main page of the app.

**Pass/Fail Criteria:** The test passes if the userid is a valid email address and the password meets the set requirements and the user registers successfully, then the user information should be successfully added to the database. Otherwise, the test fails.

**ID# 103 - Name: Retrieve Route**

**Description:** Once a user selects a particular route, the app should be able to retrieve the shortest route to the national park. Once at the destination, the user should be able to select a trail of their choice, and the app should be able to retrieve the trail route as well.

**Items covered by this test:** Route to national park, different trails at that national park.

**Requirements addressed by this test:** Functional requirements, data requirements, dependability requirements.

**Environmental needs:** Google maps API used for all the map functionalities in the app.

**Intercase Dependencies: ID# 100 -** Monitor User Location Test.

**Test Procedures:** Users should be able to select a national park of their choice from the list, the app then displays the shortest route to that national park from the user’s current location.

**Input Specification:** User presses the find route button, then selects a national park to visit. Once at the national park, the user selects a trail of their choice.

**Output Specifications:** The app should provide correct routes every time a user selects a destination.

**Pass/Fail Criteria:** If the app is able to provide correct information and routes, then the test passes. Otherwise, the test fails.

**ID# 104 - Name: Park encyclopedia**

**Description:** Users can view information about nationals parks by using the encyclopedia button.

**Items covered by this test:** Park Encyclopedia.

**Requirements addressed by this test:** Functional requirements, data requirements, dependability requirements.

**Environmental needs:** Wikipedia API used for national park information.

**Intercase Dependencies:** NA

**Test Procedures:** User presses the encyclopedia button, and then selects a park.

**Input Specification:** User presses encyclopedia button and types in a valid national park name.

**Output Specifications:** The app displays the wikipedia page for the selected national park.

**Pass/Fail Criteria:** The test passes if the app displays the correct wikipedia page for the selected park. Otherwise, the test fails.

**ID# 105 - Name: Upload Images**

**Description:** Users should be able to upload pictures from their hikes by selecting the ‘upload pictures’ button.

**Items covered by this test:** Upload pictures.

**Requirements addressed by this test:** Functional requirements, data requirements, dependability requirements.

**Environmental needs:** Firebase for picture database.

**Intercase Dependencies:** NA.

**Test Procedures:** The user presses the upload photos button, selects an image and uploads it. The photos database on firebase should have this picture stored.

**Input Specification:** The user selects a picture and uploads it.

**Output Specifications:** The picture is then visible in the picture database.

**Pass/Fail Criteria:** The test passes if the picture is successfully uploaded to the database. Otherwise, the test fails.

**ID# 106 - Name: Display Images**

**Description:** User uploaded pictures should be displayed on the app for other users to view.

**Items covered by this test:** Upload Pictures and display pictures.

**Requirements addressed by this test:** Functional requirements, data requirements, dependability requirements.

**Environmental needs:** Firebase database to store uploaded pictures.

**Intercase Dependencies:** Test ID# 105: Uploading Images

**Test Procedures:** Open the images folder and view different images.

**Input Specification:** Press the display images button.

**Output Specifications:** Display user uploaded images.

**Pass/Fail Criteria:** The test passes if users are able to view images. Otherwise, the test fails.

**ID# 107 - Name: Ask For Help**

**Description:** The program will send the data from the user’s location to the Firebase database and into the “gps\_coords” child.

**Items covered by this test:** Emergency Request Function

**Requirements addressed by this test:** GetHelp

**Environmental needs:** Location is to be set at a local park nearby in the Chicagoland region

**Intercase Dependencies:** NA

**Test Procedures:** Turn on the location service for the testing device, open the application. Log in, select the Emergency Button, hit yes on the dialog box

**Input Specification:** Hit the “Emergency” button on the main screen, then hit the “yes” button on the dialog box when it pops up

**Output Specifications:** An Android Location object is sent to the database, containing the latitude and longitude of the user

**Pass/Fail Criteria:**Pass: the gps coordinates are passed to the database. Fail: the coordinates are not passed to the database for whatever reason.

**ID# 108 - Name: Speed Test**

**Description:** The app should be fast enough to send and receive data within the desired timeframe.

**Items covered by this test:** Speed test

**Requirements addressed by this test:** Performance requirements, data requirements, dependability requirements.

**Environmental needs:** A smartphone with an internet connection.

**Intercase Dependencies:** NA

**Test Procedures:** Check all functionalities of the app, starting from user login to request help and upload photos functions.

**Input Specification:** For all user inputs, the app should send and receive information in the desired timeframe.

**Output Specifications:** The app should output desired results in the desired timeframe.

**Pass/Fail Criteria:** The test passes if the app is able to send and receive data in the desired timeframe. Otherwise, the test fails.

**ID# 200 - Name: Developer Access Only**

**Description:** The actual code should only be accessible by the developers.

**Items covered by this test:** Limited user access, developer access only.

**Requirements addressed by this test:** Security requirements.

**Environmental needs:** Firebase to access databases and SW to access backend code.

**Intercase Dependencies:** NA.

**Test Procedures:** While using the app, the user should be unable to access the original code.

**Input/output Specification:** Developers should be able to access the database through the firebase console.

**Pass/Fail Criteria:** The test passes if only the developers are able to access the code and the databases. Otherwise, the test fails.

**ID# 201 - Name: Limited Visitor Access**

**Description:** Users should not have access to the backend of the app.

**Items covered by this test:** Developer access only, limited user access.

**Requirements addressed by this test:** Security requirements.

**Environmental needs:**  Firebase to access databases and SW to access backend code.

**Intercase Dependencies:** NA.

**Test Procedures:** While using the app, the user should be unable to access the original code.

**Input/output Specification:** Developers should be able to access the database through the firebase console.

**Pass/Fail Criteria:** The test passes if only the developers are able to access the code and the databases. Otherwise, the test fails.

**ID# 300 - Name: Credible Data Provided**

**Description:** The data (routes, info about parks, current location etc.) provided by the app should be credible.

**Items covered by this test:** ID# 100 - Name: Monitor User Location, ID# 101 - Name: User Login, ID# 102 - Name: User Registration, ID# 103 - Name: Retrieve Route, ID# 104 - Name: Park Encyclopedia, ID# 105 - Name: Upload Images, ID# 106 - Name: Display Images, ID# 107 - Name: Ask For Help.

**Requirements addressed by this test:** Integrity requirements.

**Environmental needs:** NA

**Intercase Dependencies:** NA

**Test Procedures:** Login using a registered email/password, read about a national park using the encyclopedia option, find a route to that park, upload pictures from the park.

**Input Specification:** After following the above mentioned steps, the user should be able to view correct information. After logging in, the user should be logged into the correct account. After selecting a park, the user should be given correct information about the park. After uploading a picture, the user should be able to view the uploaded picture under display images.

**Pass/Fail Criteria:** If all the information provided by the app is correct, the test passes.

**ID# 301 - Name: Protect User Info**

**Description:** User information should be protected and should only be accessible by the app/developers.

**Items covered by this test: ID#200 -** Developer access only, ID#201 - limited user access.

**Requirements addressed by this test:** Security Requirements, Privacy Requirements.

**Environmental needs:** NA.

**Intercase Dependencies:** NA**.**

**Test Procedures:** Login from a user account and try to access the database.

**Input/output Specification:** User was unable to access any backend information.

**Pass/Fail Criteria:** The test passes if the user is unable to access any backend information. Otherwise, it fails.

## Test Results

**ID# 100 - Name: Monitor User Location**

**Date(s) of Execution:** 3/20/21

**Staff conducting tests:** Jasmine Gutierrez

**Expected Results:** User location should change in real time as the user moves.

**Actual Results:** User location changed in real time as the user moved.

**Test Status:** Pass

**ID# 101 - User Login**

**Date(s) of Execution:** 2/21/21

**Staff conducting tests:** Tyler Pearson

**Expected Results:** The App database should have user information for every registered user. App should be able to authenticate logins.

**Actual Results:** The App database had user information for every registered user. App was able to authenticate logins.

**Test Status: Pass**

**ID# 102 - Name: User Registration**

**Date(s) of Execution:** 3/10/21

**Staff conducting tests:** Tyler Pearson

**Expected Results:** New users should be added to the database after successful registration.

**Actual Results:** New users were added to the database after successful registration.

**Test Status:** Pass

**ID# 103 - Name: Retrieve Route**

**Date(s) of Execution:** 2/28/21, 3/5/21

**Staff conducting tests:** Mansi Chaubey

**Expected Results:** Users should be able to retrieve correct and shortest routes for the selected national parks.

**Actual Results:**  Users were able to retrieve correct and shortest routes for the selected national parks.

**Test Status:** Pass

**ID# 104 - Name: Park Encyclopedia**

**Date(s) of Execution:** 3/10/21

**Staff conducting tests:** Tyler Pearson

**Expected Results:** Users should be able to access the wikipedia page for selected national parks.

**Actual Results:** Users were able to access the wikipedia page for selected national parks.

**Test Status:** Pass

**ID# 105 - Name: Upload Images**

**Date(s) of Execution:** 4/5/21

**Staff conducting tests:** Mansi Chaubey, Jasmine Gutierrez

**Expected Results:** Users should be able to upload desired images successfully and those images should get uploaded to the database successfully.

**Actual Results:** Users were able to upload desired images successfully and those images got uploaded to the database successfully.

**Test Status:** Pass

**ID# 106 - Name: Display Images**

**Date(s) of Execution:** 4/6/21

**Staff conducting tests:** Mansi Chaubey

**Expected Results:** Uploaded images should be viewable by users.

**Actual Results:** Users were able to view uploaded images.

**Test Status:** Pass

**ID# 107 - Name: Ask for Help**

**Date(s) of Execution:** 4/01/21

**Staff conducting tests:** Tyler Pearson, Mansi Chaubey

**Expected Results:** Application would gather the GPS coordinates of the user (in this case specified to be in a local state park) and send them to the database

**Actual Results:** Same as the Expected Results

**Test Status:** Pass

**ID# 108 - Name: Speed Test**

**Date(s) of Execution:** 3/21/21, 4/1/21, 4/10/21

**Staff conducting tests:** Tyler Pearson, Mansi Chaubey, Jasmine Gutierrez

**Expected Results:** The app should send and receive data in the desired timeframe.

**Actual Results:** The app sends and receives data in the desired timeframe.

**Test Status:** Pass

**ID# 200 - Name: Developer Access Only**

**Date(s) of Execution:** 4/5/21

**Staff conducting tests:** Tyler Pearson

**Expected Results:** Developers should be the only ones who can access backend code and databases.

**Actual Results:** Only developers can access backend code and databases.

**Test Status:** Pass

**ID# 201 - Name: Limited Visitor Access**

**Date(s) of Execution:** 4/5/21

**Staff conducting tests:** Tyler Pearson

**Expected Results:** Users should not be able to access backend code and databases.

**Actual Results:** Users unable to access backend code and databases.

**Test Status:** Pass

**ID# 300 - Name: Credible Data Provided**

**Date(s) of Execution:** 4/10/21, 4/20/21

**Staff conducting tests:** Jasmine Gutierrez, Tyler Pearson

**Expected Results:** Data provided by the app should be credible.

**Actual Results:** App provides correct information, routes etc.

**Test Status:**  Pass

**ID# 301 - Name: Protect User Info**

**Date(s) of Execution:** 4/15/21

**Staff conducting tests:** Tyler Pearson, Mansi Chaubey

**Expected Results:** The database with the user info should only be accessible by the developers.

**Actual Results:** The database with the user was only accessible by the developers.

**Test Status:** Pass

## Regression Testing

* **ID# 100 - Name: Monitor User Location:** Tested once, Passed on first try.
* **ID# 101 - Name: User Login:** Repeated three times. Passed on the third try. Tests ID#200, ID#201, ID#301 repeated after test ID#101 every time.
* **ID# 102 - Name: User Registration:** Repeated twice, once after release 1 and once after release 2. Test passed on second try. Tests ID#200, ID#201, ID#301 repeated after test ID#102 every time.
* **ID# 103 - Name: Retrieve Route:** Tested once, Passed on first try.
* **ID# 104 - Name: Park Encyclopedia:** Tested multiple times throughout the development of the project. ID#300 tests repeated after this test every time.
* **ID# 105 - Name: Upload Images:** Tested multiple times. Tests ID#200, ID#201, ID#301 repeated after test ID#105 every time.
* **ID# 106 - Name: Display Images:** Tested once, Passed on first try.
* **ID# 107 - Name: Ask For Help: Tested multiple times.** Tests ID#100, ID#103, ID#108 repeated after test ID#107 every time.
* **ID# 108 - Name: Speed Test:** Tested multiple times throughout the development process.
* **ID# 200 - Name: Developer Access Only:** Tested once, Passed on first try.
* **ID# 201 - Name: Limited Visitor Access:** Tested once, Passed on first try.
* **ID# 300 - Name: Credible Data Provided:** Tested multiple times throughout the development process.
* **ID# 301 - Name: Protect User Info:** Tested once, Passed on first try.

# Inspection

Second Button in The List Activity:

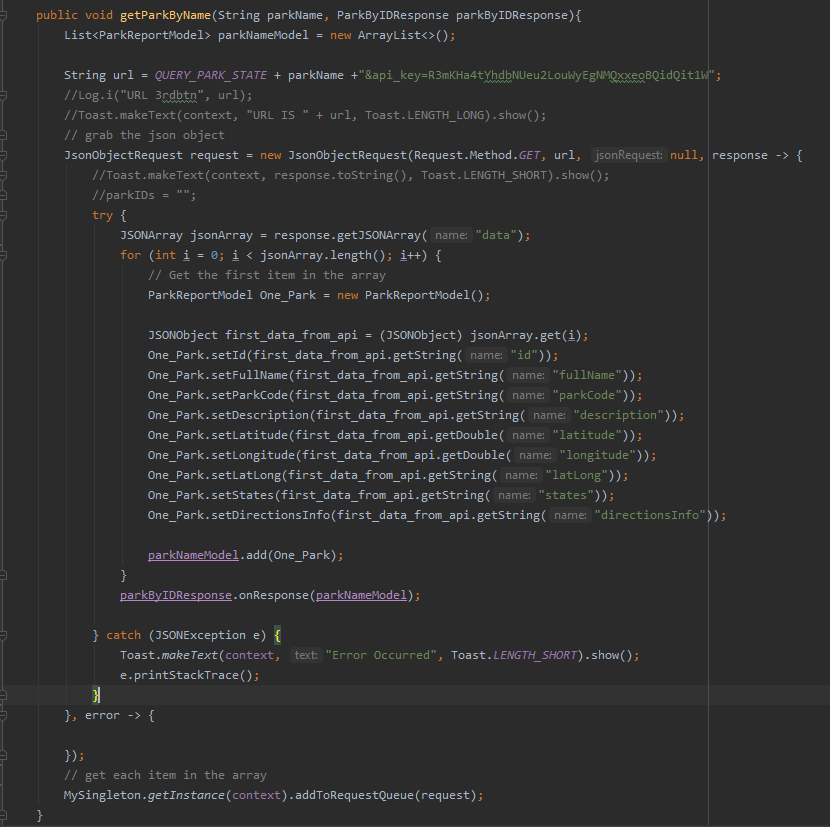
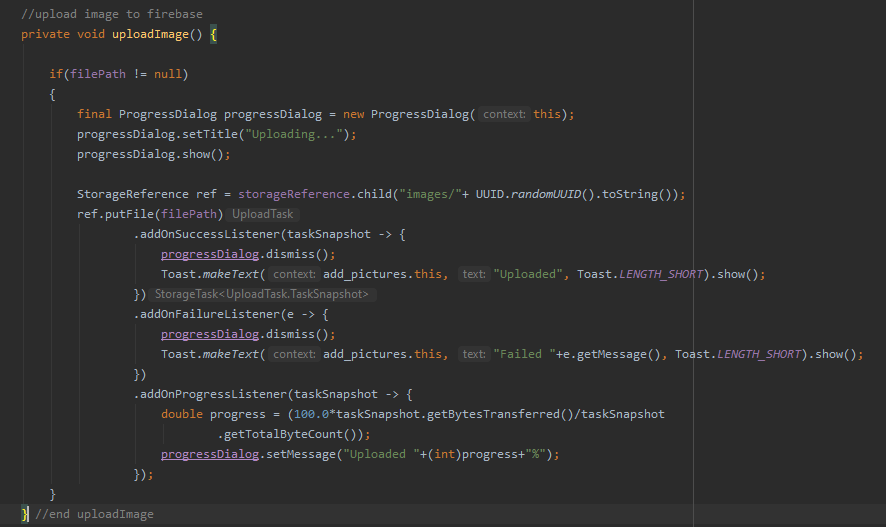
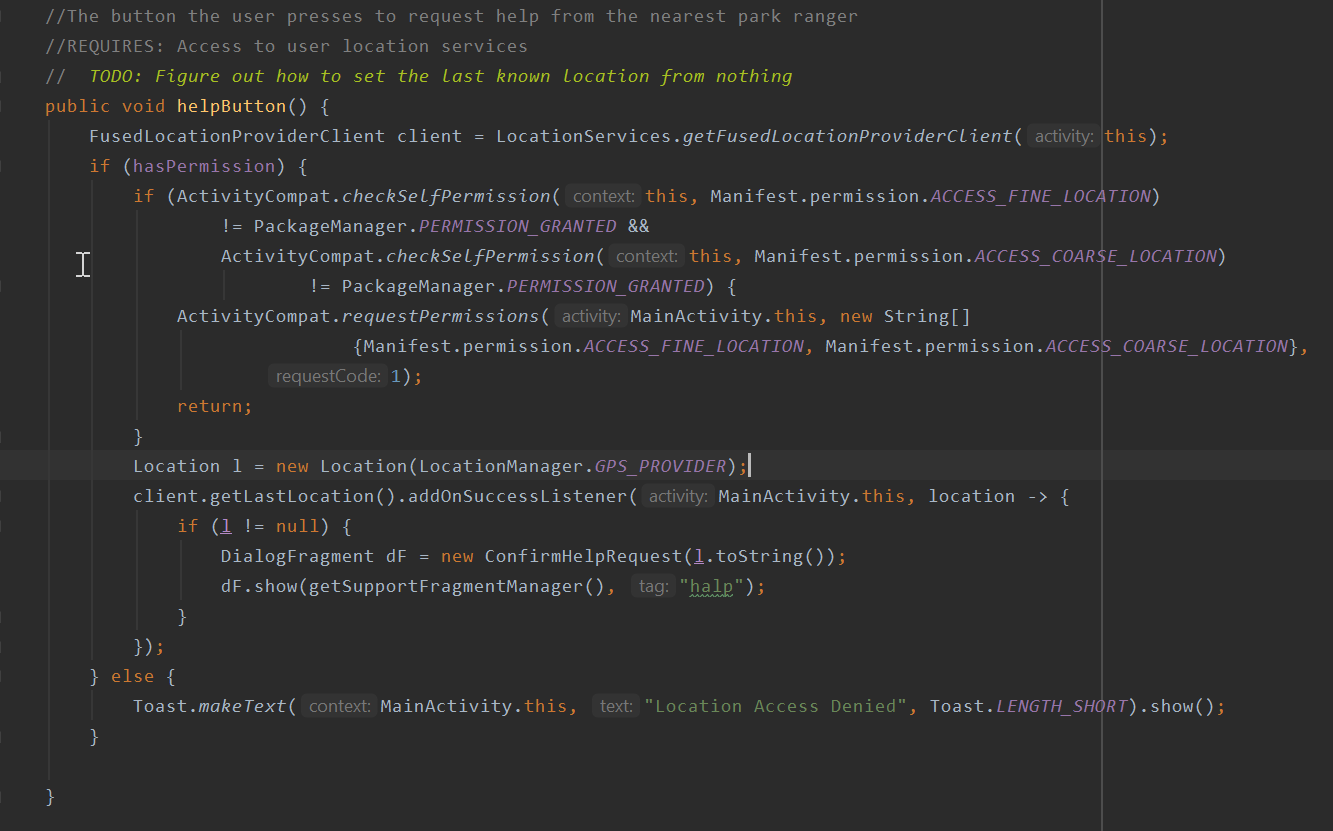


Image Activity Code:



helpButton():



## Items to be Inspected

* btn\_getLongLangByID() in List Activity
* uploadImage() in Upload Image Activity
* helpButton() in Main Activity

## Inspection Procedures

[Glossary](#_23ckvvd)

## Inspection Results

Mansi: Tested encyclopedia API

Conclusions from testing: Mansi concluded that the encyclopedia does return results for certain aspects and information about parks and their history as well as info of flower species and other plant species. The search did not consider that a user may search other things that could be random, so that was not considered to filter out other topics that are different then the nature parks.

Tyler: Tested the Route API

Conclusions from testing: Route API works as intended with no issues present during inspection. However, the return results return park info with an parkID rather than other information of the park that would be more reasonable, such as the park name. Overall, the fetch using volley works with the parksIDs and returns the correct information. However,the code should request a user to enter something more easier for them, such as the name.

Jasmine: Tested Picture Functionality

Conclusions from Testing: Jasmine tried to add different types of files to trip up the code when it gets sent into a firebase database, but it did not work. The code was implemented correctly. She was able to send in pictures as well as take pictures at that instant, and the pictures both appeared on the database.

# Recommendations and Conclusions

The items listed above have passed their inspections and tests. The next idea to move forward with is to begin further implementation and more testing as needed.

# Project Issues

## Open Issues

The project did not achieve development on multiple platforms due to time and resources constraints. In the future, if the project is picked up again by members of the development team, making versions for different platforms should be the first objective to complete. Bare minimum, these new, non-Android versions should be on the same level functionality wise as the Android version currently is as of the writing of this document.

## Waiting Room

N/A

## Ideas for Solutions

When a development team needs to create a program that is cross platform, i.e. needs to be on both iOS and Android, the team can use multi-platform development technologies like Flutter or Xamarin to do so, rather than trying to develop them in their own IDEs individually.

## Project Retrospective

For future reference, a better understanding of how merges for git work would have saved the development team a lot of time and headaches solving the issues presented by them, as well as creating separate branches for each member’s code and then merging branches as necessary, rather than pushing everything into a single master branch at a time.

More concrete development schedules are definitely necessary for this kind of project, and a better schedule would have helped just as much as more efficient uses of git would have. Implementing the schedule via a calendar app would have been a massive boon to productivity for this project.

In reference to multi-platform development, it would have been much more useful to use the services listed in the glossary or others as available to not have to worry about how to make multiple versions of the same project. Using them (and knowing they existed in the first place) would have allowed the development team to fulfill one of the main focuses of the design team’s document[5]: an iOS app.

# Glossary

**Park:** refers to National Parks

**Phone:** refers to smartphones

**Interactive Map:** references a map that adjusts to a current user’s location. Increases the cost of the product due to having to implement it into the product with a license

**JUnit**: The standard testing framework used in Java development and used specifically in this project

**androidx.test**: API used in tandem with JUnit with making and running the smaller tests

**Espresso**: API used in tandem with JUnit and androidx.test to make and run the larger tests on the product, especially with the UI elements in the project.

**Firebase:** Account authentication/recovery/database service used in the development of the project; made by Google

**Flutter:** Multi-platform development IDE designed by Google that uses the Dart programming language

**Xamarin:** Multi-platform development IDE designed by Microsoft that uses C# and .NET programming languages

**Volley:** A HTTP Library that makes networking for Android Apps Easier

# References / Bibliography

|  |  |
| --- | --- |
| **[1]** | **Robertson and Robertson, Mastering the Requirements Process.** |
| **[2]** | **A. Silberschatz, P. B. Galvin and G. Gagne, Operating System Concepts, Ninth ed., Wiley, 2013.** |
| **[3]** | **J. Bell, "Underwater Archaeological Survey Report Template: A Sample Document for Generating Consistent Professional Reports," Underwater Archaeological Society of Chicago, Chicago, 2012.** |
| **[4]** | **M. Fowler, UML Distilled, Third Edition, Boston: Pearson Education, 2004.**  **[5] Chau, Tony, et al. University of Illinois at Chicago, 2020, pp. 1–48, *Lone Wolf National Park App* .**  **[6]** **“Visitation Numbers (U.S. National Park Service).” National Parks Service, U.S. Department of the Interior,** [**www.nps.gov/aboutus/visitation-numbers.htm**](http://www.nps.gov/aboutus/visitation-numbers.htm)**.**  **[7] “How Much Are National Parks Worth to You?” *SOURCE*, 16 Mar. 2018, source.colostate.edu/how-much-are-national-parks-worth-to-you/.** |

# Index

**No index entries found.**