Info Parks USA

Group 30
Prepared by
Hasan Ali, Umer Qazi, Syed Raza and Claudio Jimenez
***for use in* CS 440**
at the
University of Illinois Chicago

Fall 2020

Table of Contents

Project Description

Project Overview

Info Parks USA is a mobile application design to help the user planned for a trip and access information of any US National Park. The application main features are,

1. A packing List: The system provides the user with a tailored packing list depending on the dates and park he is visiting.
2. Find camping grounds: The system is able to find the camping grounds around the user depending on his phones GPS location. Very useful feature while in the park.
3. Get information about the park: The system provides the user with,
    a. A map of the park
    b. The parks website
    c. Weather information about the park

The Application will be developed in conjunction with the US National Park Services and will be release for Android and iOS.

Project Domain

Domain was developing an android application using java. This was done using an MVC architecture. We used google chrome and maps API for location and web services. We also used 10 activities to implement the different features of the application. We tested our application using a android virtual emulation running Android 9.0 API 28

Relationship to Other Documents

The document **we got the idea of** this **app was called Info Parks USA written by Group 5 of the Spring 2020 semester. Bumjargal Boldbaatar, Elijah Mangaba, Jared Manusig, and Mahar Basrawi were the original authors of this application idea. We were able to find the blueprint for this** project **with explanations for all the small details as well. Requirements and Design were very important for our implementation of the project.**

Naming Conventions and Definitions

- We used camel case for class names

Definitions of Key Terms

Application: Refers to the Android Application of InfoParks USA

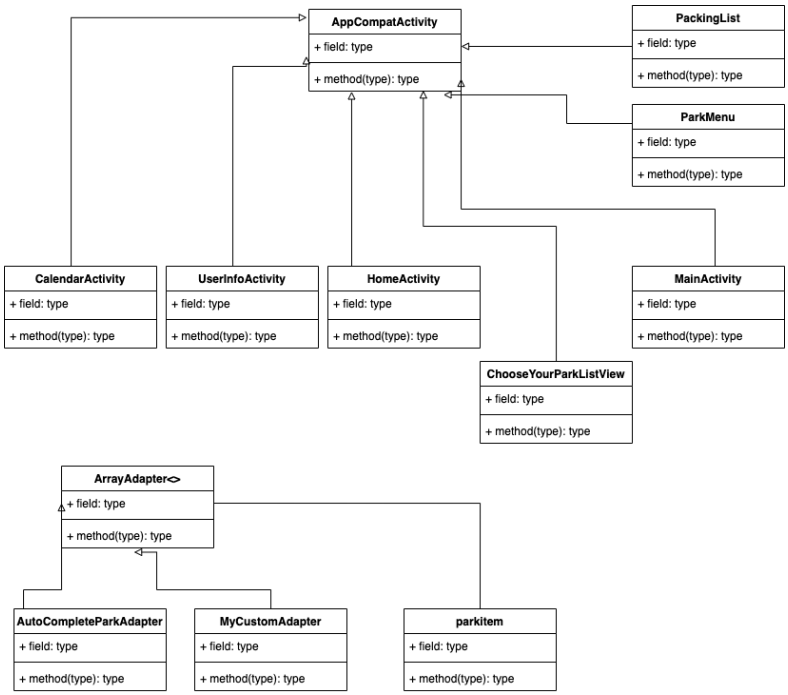
Packing List: In context of this app, refers to having multiple packing lists for the different months

Camping Ground: The feature of being able to locate camping grounds near the user

UML and Other Notation Used in This Document

This document follows the Version 2.0 OMG UML Standard.

UML diagram of the Application:



Data Dictionary for Any Included Models

User Base = The number of users that have an account in the Application

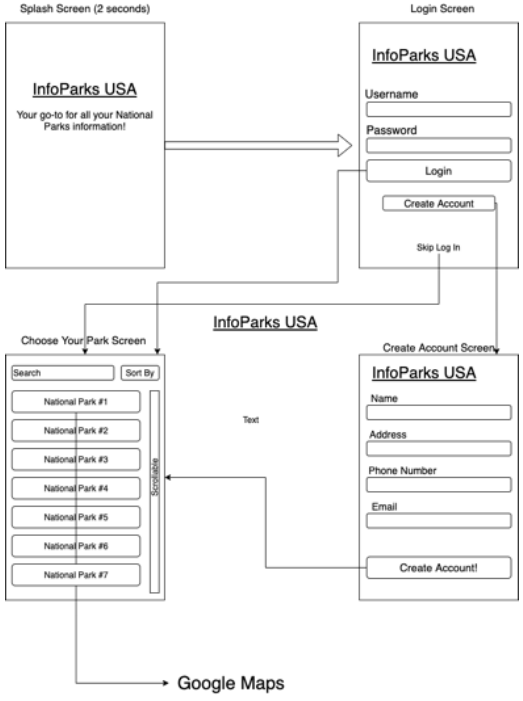
Project Deliverables

We have produced an Android application with functionality to explore the various National Parks in the United States. The application gives the user access to different important information regarding the Park as well as keeping track of any items they would like to take through the use of a packing list. Along with the ability to add and remove items, a suggested list of items is offered to the user based on the month that they are traveling and the corresponding weather in that month.

First Release

Date: 11/13/20

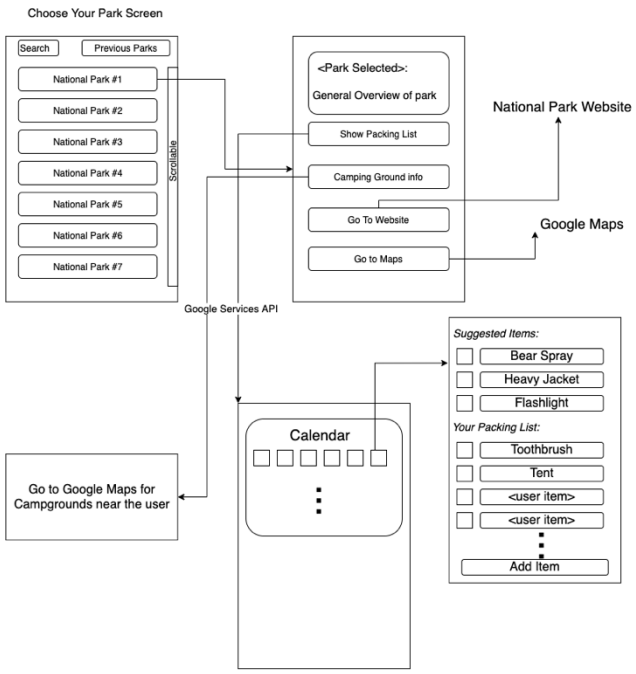
The system allowed users to login or create a new account if this was their first time using the application. Once the user, selected the relevant option for them, a list of parks was presented to the user and they were expected to select the park they were interested in. When the user selects the specific park, they would be redirected to the 'Google Maps' application where they would be able to see a multitude of different options such as navigation and attractions nearby.



Second Release

Date:

The system allows the user multiple options after they have selected a park to explore. They can go explore the maps as was mentioned in the first release. Along with that, they had three other options which were Web, Camping Ground and Packing list. With the web option, they were able to explore the specified parks website for more information. For the camping ground, the application picked up the location of the user and showed them close camp grounds in Google Maps. Then, as for the last option, once the user selected the packing list, they were asked to input the date that they are traveling. Once that was inputted, they were presented with a list of suggested items that they could remove as well as the ability to add (and remove) their own items.



Comparison with Original Project Design Document

The original authors envision an application that combined information services with GPS location. The system will be able to pinpoint the user's location and provide info about the parks weather, main attractions, alerts, maps, camping grounds, businesses and what to pack for your trip.

Our high-fidelity prototype implemented the majority of the previously mentioned services, including a packing list feature. We followed the constraints on user access and user creation envisioned by the authors. The prototype also implemented the "find camping grounds" feature to show the full potential of the application info services pair with the phone's GPS location.

Testing

Items to be Tested

- The functionality of the packing list
- The functionality of different packing lists coming up based on the weather of the month the user is traveling
- The ability of the application to pick up the accurate location of the user
- The Application providing the correct link to Google Maps for the park that is selected by the user

- The Application providing the correct link to the website for the park that is selected by the user
- The application not allowing entry into the application until an account is made or the user logs in
- Successful input of a new user into the user database
- Ability for the application to recognize a user that is already in the database

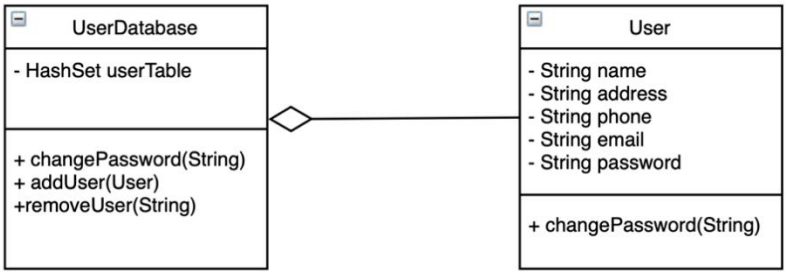


Fig. 4b.1 User and User Database class diagram

Test Specifications

ID# - Name: UserUnitTest

Description: Unit test for the User class

Items covered by this test: User class

Requirements addressed by this test: Requirement #01 and Requirement #03

Environmental needs: Use Android Studio testing environment to run the tests.

Inter-case Dependencies: None

Test Procedures:

1. Open InfoParksUsa.app in Android studio as an existing project.

2. From the project view, go to app/java/com.CS440.infoparkusa(test)

3. Choose run test in ‘info park usa’ from run menu.

Input Specification: All inputs for the test are strings. This is an automated unit test; inputs have been pre-populated before. You can change the inputs for the test by changing the values of the data members of the test class.

Output Specifications: The automated unit tests should all pass, and you should see green checkmarks on all the unit tests in the testing pane of Android Studio.

Pass/Fail Criteria: All tests will be passed if the user object was properly created with all of the data fields completed, otherwise the tests will fail. See unit Tests file for test details.

ID# - Name: UserDatabaseUnitTest

Description: Unit test to test the UserDatabase class

Items covered by this test: User database class

Requirements addressed by this test: Requirement #01 and Requirement #03

Environmental needs: Use Android Studio testing environment to run the tests.

Inter-case Dependencies: Run UserUnitTest test first.

Test Procedures:

1. Open InfoParksUsa.app in Android studio.

2. From the project view, go to app/java/com.CS440.infoparkusa(test)

3. Choose run test in ‘info park usa’ from run menu.

Input Specification: All inputs for the test are User objects. This is an automated unit test; inputs have been pre-populated before. You can change the inputs for the test by changing the values of the data members of the test class.

Output Specifications: The automated unit tests should all pass, and you should see green checkmarks on all the unit tests in the testing pane of Android Studio.

Pass/Fail Criteria: All tests will be passed if the user object was properly added to the user database, otherwise the test will fail. See unit Tests file for test details.

ID# - Name: PackingListAddition

Description: Test for adding the item typed by the user into the packing list

Items covered by this test: PackingList class

Requirements addressed by this test: Requirement #08

Environmental needs: Run the Android Emulator to see it in real-time

Inter-case Dependencies: N/A

Test Procedures:

1. Open InfoParksUsa application in Android studio.
2. Run Android Emulator
3. Navigate to packing list and ensure that items can be added and viewed.

Input Specification: The input is whatever valid text the user would like to save in their packing list

Output Specifications: The added item should be shown on the Android Emulator screen and the option to remove it should be there as well

Pass/Fail Criteria: All tests will passed if the user is allowed to add as many items as they would like and to instantaneously view them in the packing.

ID# - Name: PackingListRemoval

Description: Test for removing an item in the packing list

Items covered by this test: PackingList class

Requirements addressed by this test: Requirement #08

Environmental needs: Run the Android Emulator to see it in real-time

Inter-case Dependencies: N/A

Test Procedures:

1. Open InfoParksUsa application in Android studio.
2. Run Android Emulator
3. Navigate to packing list
4. Delete one of the suggested items and/or input your own and then delete it

Input Specification: The input is the pressing of the 'remove' button by the user

Output Specifications: The output is the deletion and removal of the item form the screen

Pass/Fail Criteria: All tests will passed if the user is allowed to delete as many items as they would like

ID# - Name: MapsConnectionforParks

Description: Test for connecting the map to the specific park picked by the user

Items covered by this test: ChooseYourParkListView class

Requirements addressed by this test: Requirement #7

Environmental needs: Use Android emulator to test in real time

Inter-case Dependencies: None

Test Procedures:

1. Open InfoParksUsa application in Android studio.
2. Run Android Emulator
3. Go to Map option and ensure google maps is properly showing the designated park

Input Specification: Input is pressing the 'Map' option after selecting park

Output Specifications: The output is transferring to Google Maps and .

Pass/Fail Criteria: All tests will be passed if the app correctly takes the user to google maps showing the correct park. Otherwise, fail.

ID# - Name: WebsiteConnectionforParks

Description: Test for connecting the website to the specific park picked by the user

Items covered by this test: User class

Requirements addressed by this test: Requirement #09

Environmental needs: Use Android Studio testing environment to run the tests.

Inter-case Dependencies: None

Test Procedures:

1. Open InfoParksUsa application in Android studio.
2. Run Android Emulator

3. Go to Website option and ensure the park site is properly showing the designated park

Input Specification: Input is pressing the 'Website' option after selecting park

Output Specifications: The automated unit tests should all pass, and you should see green checkmarks on all the unit tests in the testing pane of Android Studio.

Pass/Fail Criteria: All tests will be passed if the app correctly takes the user to showing the correct park website. Otherwise, fail.

ID# - Name: PackingListBasedOnWeather

Description: Testing if app provides proper packing list based of date of trip

Items covered by this test: Calendar Activity & Packing List classes

Requirements addressed by this test: Requirement #10 and Requirement #13

Environmental needs: Use Android Studio testing environment to run the tests.

Inter-case Dependencies: None

Test Procedures:

1. Open InfoParksUsa application in Android studio.
2. Run Android Emulator
3. Select Date of Winter and check if packing list shows coat
4. Select date of summer and check if packing list shows bug spray

Input Specification: Input would be the date selected

Output Specifications: Output would be the packing list that shows up based on the date.

Pass/Fail Criteria: All tests will be passed if the packing list shows coat for winter date and bug spray for summer date. Otherwise fail.

Test Results

ID# - UserUnitTest-Result

Staff conducting tests: Syed Raza

Expected Results: User should be created with the given parameters

Actual Results: User was successfully created with the given credentials

Test Status: Pass

ID# - UserDatabaseUnitTest-Result

Staff conducting tests: Syed Raza

Expected Results: Said user should be added to the user database

Actual Results: The user was added to the database after the test was run

Test Status: Pass

ID# - PackingListAddition -Result

Staff conducting tests: Hasan Ali

Expected Results: The same text string added in the EditView is the one in the packing list

Actual Results: Same text string added to the list with the option to remove it

Test Status: Pass

ID# - PackingListRemoval -Result

Staff conducting tests: Hasan Ali

Expected Results: The item with the associated remove button is taken out of the list

Actual Results: Item was removed from the list

Test Status: Pass

ID# - WebsiteConnectionforParks -Result

Staff conducting tests: Claudio Jimenez

Expected Results: Website option sends user to the parks website for more information

Actual Results: Website is opened in the user's preferred browser for the correct National Park

Test Status: Pass

ID# - MapsConnectionforParks -Result

Staff conducting tests: Claudio Jimenez

Expected Results: Map option sends user to the Google Maps application for navigation, etc.

Actual Results: User is sent to Google Maps for the National Park of their choice for more information

Test Status: Pass

ID# - PackingListBasedOnWeather -Result

Staff conducting tests: Umer Qazi

Expected Results: The suggested packing list changes based on the month user selected

Actual Results: Packing list changes to the specific weather associated with said month

Test Status: Pass

Inspection

Items to be Inspected

- The functionality of the packing list
- The functionality of different packing lists coming up based on the weather of the month the user is traveling
- The ability of the application to pick up the accurate location of the user
- The Application providing the correct link to Google Maps for the park that is selected by the user
- The Application providing the correct link to the website for the park that is selected by the user
- The application not allowing entry into the application until an account is made or the user logs in

Inspection Procedures

- **Packing List**
    - Check for addition and removal functionality
    - Check for changing suggested list based on month
- **Camping Ground**
    - Ensure user location is being picked up accurately
    - Successfully switching application to Google Maps
- **Maps**
    - Successfully connects to Google Maps on the user's phone
    - Ensure map shows the park that the user selected
- **Website**
    - The user is directed to the website specific to the park that the user selected
- **Meetings**
    - Two meetings weekly
    - One designated member to write meeting notes and upload on Github
- **Jira**
    - Add stories and subtasks per sprint
    - Divide work and assign story minutes

Inspection Results

Packing List

What was inspected: Check to see if each weather has different items.

Who did the inspection: Umer Qazi, Claudio, Hasan

Result of inspection: Pass

Notes: The winter list would get heavy jackets and spring would get umbrella.

Camping Ground

What was inspected: The camping ground would show the map of the park with potential areas for camping.

Who did the inspection: Umer, Hasan, Syed

Result of inspection: Pass

Notes: The map shows up with areas of hiking and camping.

Maps

What was inspected: The map needs to show the map of the park when requested and when hit camping ground; it needs to show the surrounding map and not the whole park.

Who did the inspection: Hasan, Syed, Claudio

Result of inspection: Pass

Notes: It shows the camping ground near where it was searched.

Website

What was inspected: Connecting and getting information from the national park website.

Who did the inspection: Umer, Syed, Claudio

Result of inspection: Pass

Notes: The website would return information about the specific park that was searched.

Meetings

What was inspected: The timings and if they were productive.

Who did the inspection: Umer, Syed, Claudio, Hasan

Result of inspection: Pass

Notes: We all were mostly presented on time and the meetings were productive. We used Discord server. Meetings were mostly led by Claudio.

Jira

What was inspected: The process of ticketing is smooth and effective.

Who did the inspection: Umer, Syed, Claudio, Hasan

Result of inspection: Pass

Notes: The process to pass on the tickets and tasks worked very smoothly

Recommendations and Conclusions

Project Issues

Open Issues

- Our beta release yielded some feedback on the project. The application could improve dramatically by adding a multi-park trip planning feature.
- Our security consultant express concern with new regulation on privacy laws coming from congress.

Waiting Room

Some ideas in mind we had was revolved on enhancing the users experience on the app. In particular, we would like to work on specific activities to do in the park. This would make each park experience unique and allow the user to have an idea on what each park has to offer. Another idea we would like to add is having a chat system that lets other campers leave reviews and recommendations based on each park they visited. This will give campers an idea of the pros and cons of each park. Lastly, we would like to add a feature review for the app where the user sees a pop up screen asking if they would like to review the app and let us developers know what the users enjoy and what opportunities we can fix for the future.

Ideas for Solutions

Connection with our APIs: More connection with different APIs for ease of use so we can benefit from more national park APIs.

Apple version: Our app is currently in Android but making it in an iPhone friendly environment will help us reach more users.

Project Retrospective

Communication: we used Discord, and this was very helpful as everything was centralized in one place. For example, there were different channels for the coding and development project, there was a voice channel for meetings, and we could have shared any files, etc. with all of the team members. In that regard it was very productive.

On the other hand, it is to be expected that there will be communication struggles so what helped us was to have two set times for meeting and ensure that if somebody had to postpone, they try to do it at the very least two days before.

If the current online learning setup continues, then communication would easily be the biggest hurdle that a group has to overcome. It is important for everybody to be on the same page and make sure that everybody is pulling their weight and doing all of the expected work.

Organization: Regarding weekly minutes, it is good to have one person that is dedicated for a couple tasks on a weekly basis. For example, in our case, there was one person in charge of taking notes for the meeting, uploading them to GitHub, and starting the next sprint on Jira for the next week. This way the responsibilities are well defined and everybody is on the same page.

References / Bibliography

Bell, J. (2012). *Underwater Archaeological Survey Report Template: A Sample Document for Generating Consistent Professional Reports.* Chicago: Underwater Archaeological Society of Chicago.

Fowler, M. (2004). *UML Distilled, Third Edition.* Boston: Pearson Education.

Robertson, & Robertson. (n.d.). *Mastering the Requirements Process.*

Silberschatz, A., Galvin, P. B., & Gagne, G. (2013). *Operating System Concepts* (Ninth ed.). Wiley.

Index

No index entries found.