**GitHub Username**: sbj1811

National Park Guide

# Description

National Park Guide boasts rich, detailed information on America’s most popular National Parks, including key features, history, maps, campgrounds, hiking trails, current weather and alerts, with links to additional information resources using [NPS Data API](https://www.nps.gov/subjects/digital/nps-data-api.htm) , [Hiking Project Data API](https://www.hikingproject.com//data) and [OpenWeatherMap API](https://openweathermap.org/api).

# Intended User

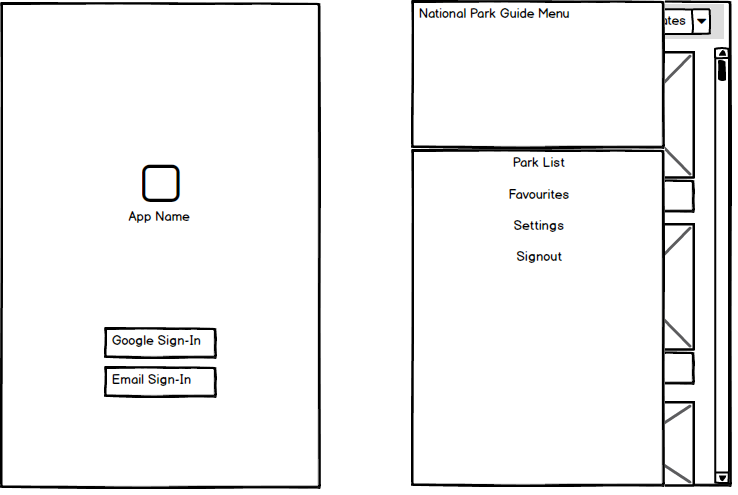
The application is aimed at Tourists and Hikers visiting any of the National Parks in United States of America.

# Features

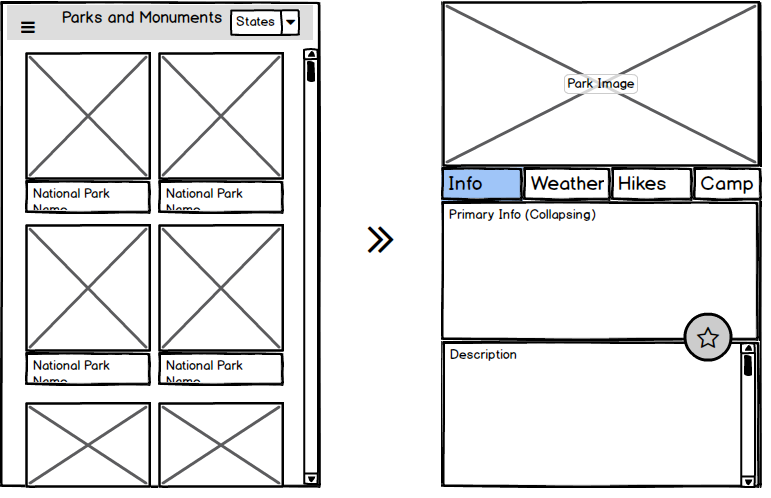
* Descriptions of each park’s key features and history, with links to additional resources.
* Photo gallery stocked with high-quality park images
* Push notifications for park-related news, alerts, and events.
* Tag your favorite park sights.

User Interface Mocks

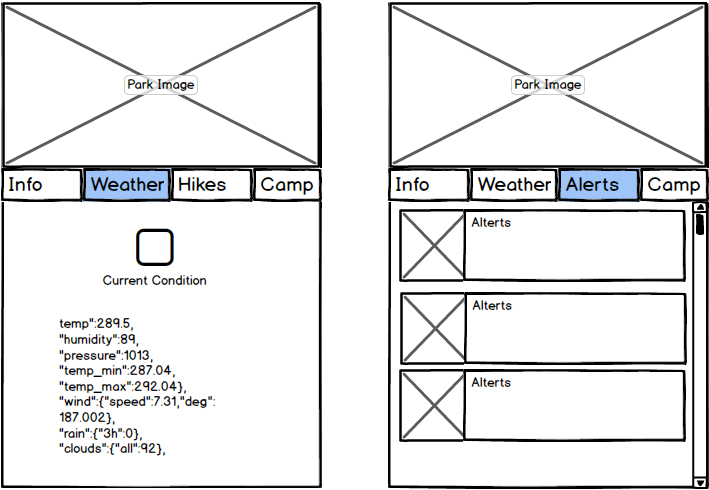
**Login and Menu**



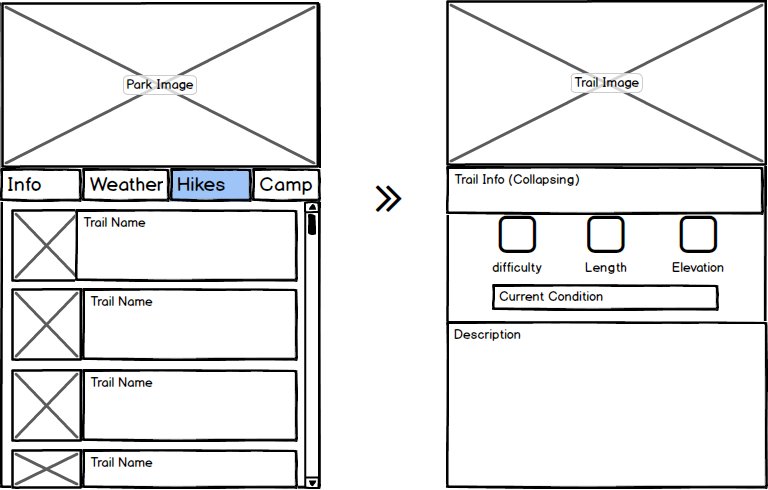
## Main Menu and Detailed Info View



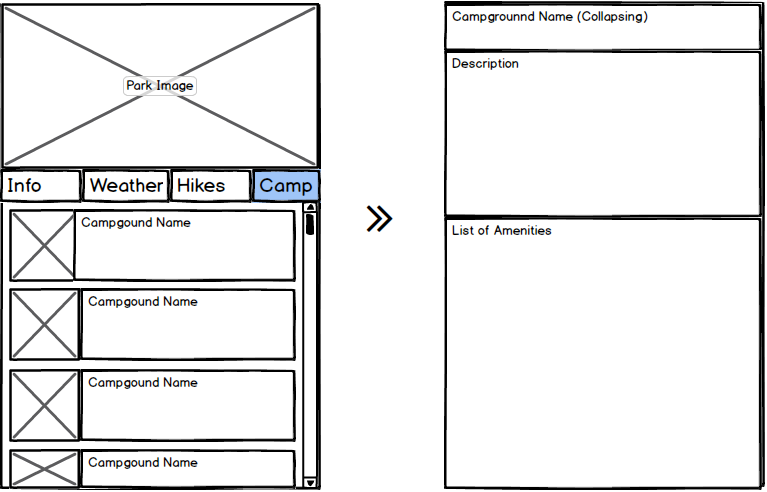
## Current Weather Condition and Alerts



**Hiking Trails List and Detailed Info View**



## Campground List and Detailed Info View

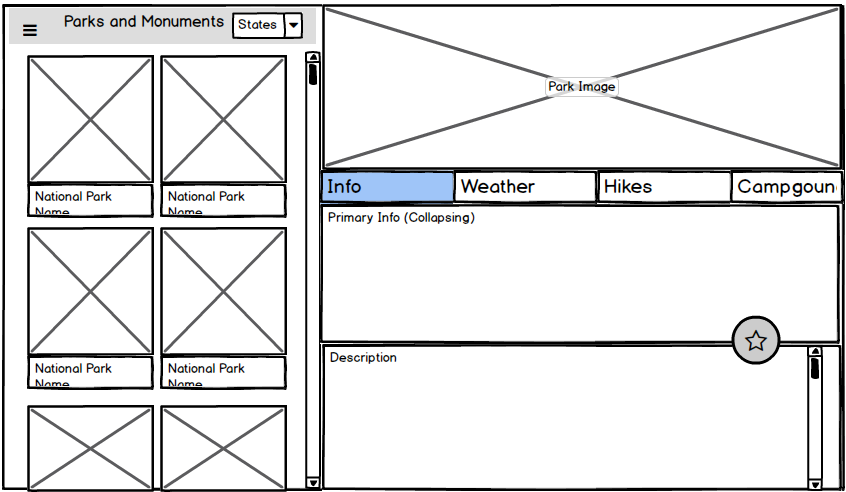


## 

## 

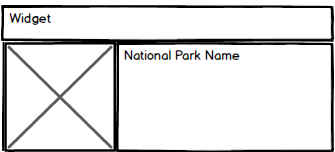
## 

## Tablet Layout



# 

# Widget



# 

# Key Considerations

### How will your app handle data persistence?

App will use a Content Provider and a Shared Preferences to maintain the local data.

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

The application must not crash in unstable or missed network connection cases.

The application must handle all long running operations correctly considering possible configuration changes.

The application must not use the main thread for any resource consuming operations.

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

Glide: Loading and caching of images.

Retrofit 2: Network API request

Butterknife: reducing Boilerplate code

Stetho: Debug

Firebase Realtime Database

Firebase Analytics

Google Places API

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

This application will use Firebase Realtime Database, Firebase Analytics and Google Places API.

# Next Steps: Required Tasks

## Task 1: Project Setup

Creating a new project in Android Studio v3.1.2

Configuring libraries by adding all necessary dependencies using Gradle v4.4

App will be written solely in the Java Programming Language

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

* Build UI for MainListActivity
* Build UI for ParkInfoActivity
  + Implement Weather Fragment
  + Implement Alert Fragment
  + Implement Hikes Fragment
  + Implement Events Fragment
  + Implement Campground Fragment
* Build UI for MapView

## Task 3: Implement Network Sync for all API services

* Fetch data using Retrofit2, SyncAdapter for NPS API and AsyncTask for other APIs.
* Create data classes which help to handle all response data provided by API calls.

## Task 4: Local Storage and Firebase Realtime Database

* Implement Google login for storing user’s favorites in Firebase Realtime database.

## Task 5: Implement Widget

* Implement widget for Favorite national park.
* Build UI for widget
* SQLite Local storage/Shared Preference for displaying information on Widget
* Uses ContentProvider and Loader to move data to its views

## Task 6: Polish UI

* Modify layout for both phone and tablet
* Visualize historical data of cryptocurrency
* Design and implement the theme
* Polish layout, color, font, etc
* Using material design features if needed

## Task 7: Testing

* Write basic testing cases using Junit and Expresso
* Fix bugs if needed

**Task 8:** Take the app production ready

* Improve the app visuals using some Material Design techniques (shared element transitions, parallax scrolling, animations etc.)
* Provide RTL support.
* Provide accessibility support.
* Provide content descriptions for meaningful UI elements
* Manage resources and plan for localization (move all strings into strings.xml)

**Task 9: P**repare for release

* Create app icon (square and round variations)
* Remove or disable debug messages and info
* Follow steps from Android Developers Launch Checklist.