GitHub Username: baksoy

WhatPlace

Description

WhatPlace let's the user add a place by selecting from a list of Google Places nearby the user's current area. The user then saves the place they are interested in on their phone to come back to it later.

Often times we are interested keeping information of places we either visited or want to visit later. This app keeps a list of such places by letting the user keep pertinent data of such places.

Intended User

This app can be useful for multiple profiles of people. Salespeople can be scouting an area to look for what businesses are in the area and pull their name, address, website, and phone number.

People who like to dine out can keep list of their favorite places to eat.

Travelers or commuters who are not familiar with a particular area and would like a quick discovery of the area in a jiffy.

Anyone who is interested in keeping a list of places they want to save for reference

Features

List of WhatPlace features:
- Detects current location of user
- Let's user search nearby places
- Saves information of a chosen place
- Upcoming features in future releases:
  - Take pictures
  - Search with autocomplete
  - Keeps track of favorite places
  - Geofence notification where user is notified when they are in certain proximity of their favorite place
  - One-click launch of launch of driving directions to a place

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.
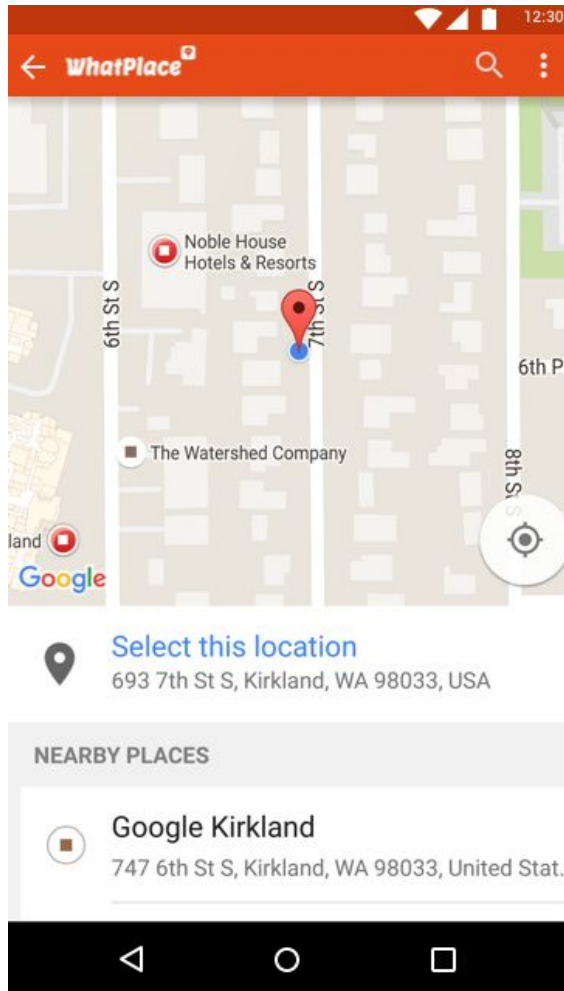
Screen 1

No place added yet.
Press the add button to see a
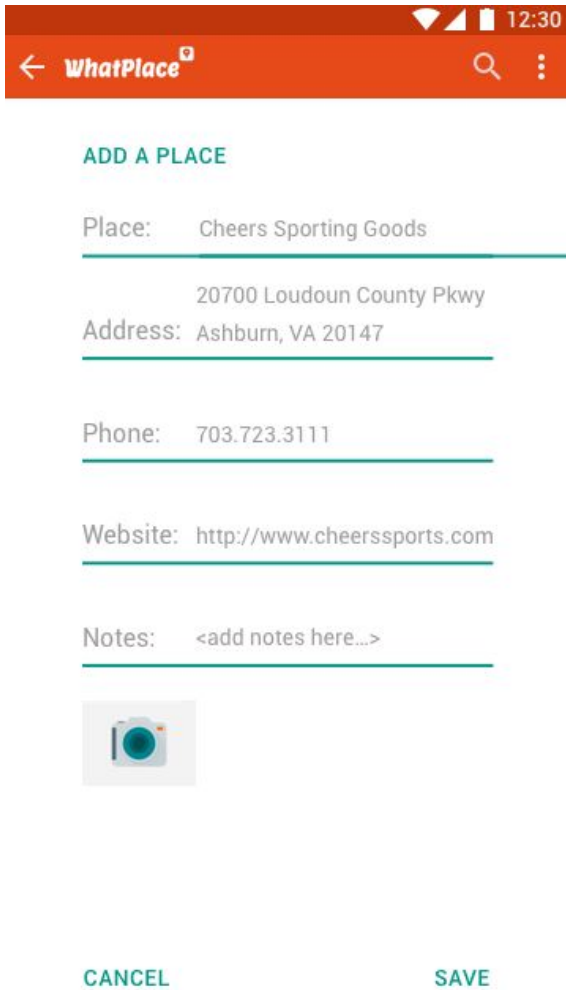list of nearby places and
select a place.

Initial screen when the user first launches. The app let's the user know they didn't add a place yet and instructs them to press the Fab button to start adding their places of interest.

Screen 2

Clicking the Fab button in the initial screen launches the Places Picker screen, detects the user's location, and shows on map Google Places nearby both as a list and icons on the map.

Screen 3

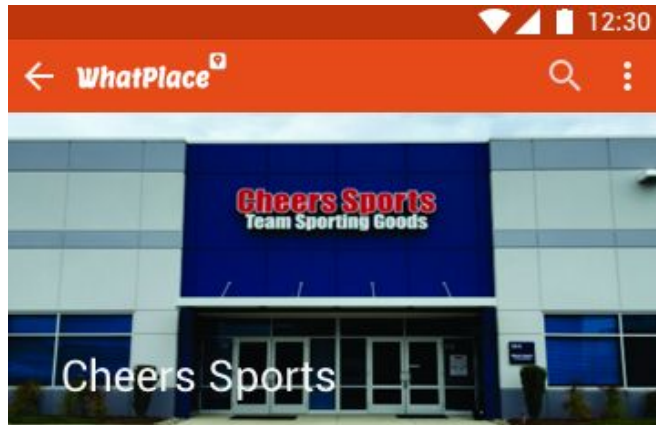Once the user has selected a place on the map in the previous screen, the app moves to the next screen (PlaceAddActivity), auto filling in the form's place name, address, phone number, and website. The user can add notes and take a picture if they wish and click save or cancel if they decide to back out.

Screen 4

Once the user has saved the Place in the previous screen, the app lands them on the PlaceDetailActivity screen showing them the details of the Place they saved. The user can move from here to the main screen by clicking the back arrow on the toolbar or they can chose to edit the information by clicking Edit button.

Screen 5
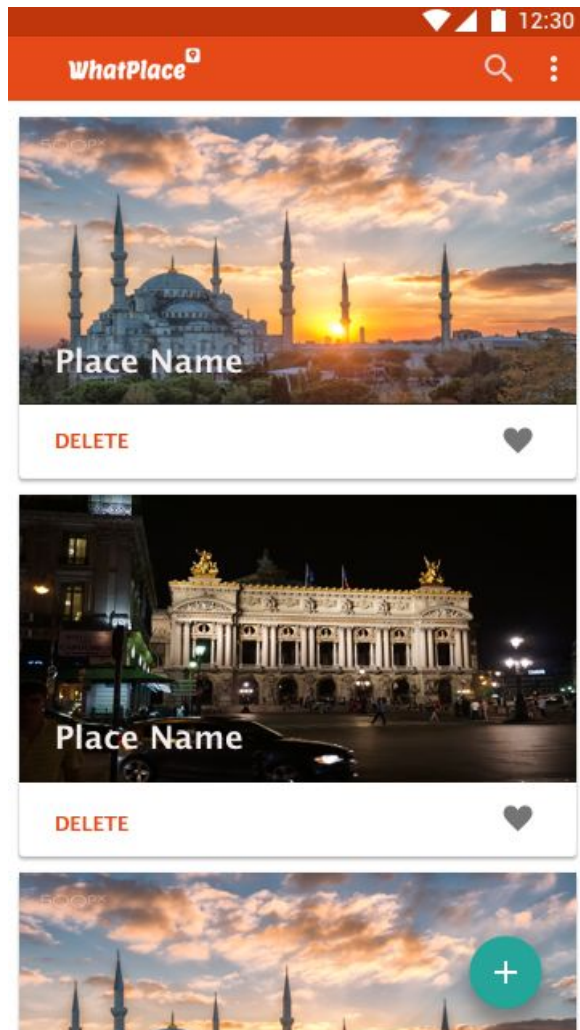
This is the Main screen once the screen is populated with data showing CardView list of Places. The user can click the add Fab button to add more places.

Key Considerations

How will your app handle data persistence?
Describe how your app with handle data. (For example, will you build a Content Provider or connect to an existing one?)
Data will be saved to the phone's local storage. Handling data will be implemented via the ContentProvider API. I will be building a new Content Provider for What Place.

Describe any corner cases in the UX.

For example, how does the user return to a Now Playing screen in a media player if they hit the back button?

If the user backs out of a form without saving the data, they will get an Alert Dialog asking "Are you sure you want to exit without saving the data" and ask for confirmation to leave that screen without persistence.

If the user doesn't have data connection they will be asked to turn it on for adding new places or launching the map.

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

Butterknife to make the code less verbose by view binding, resource binding, and listener bindings with annotations.

Glide (if/when I integrate pictures) will be used for image loading because it is recommended by Google and it makes development easier.

RecyclerView and **CardView** will be used for efficient because of its ability to display large data sets that can be scrolled very efficiently by maintaining a limited number of views.

Google Play Services will be used because it is required for using Google Places API and AdMob. Places API is what allows the app to pull all nearby places from the Google repository. AdMob library is used for monetization via advertisements.

Android Design Support Library will be used to adopt Material Design look and feel (theme and animations).

AppCompat will be used because the Design library depends on it and also for backward compatibility.

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

Initial steps to setup project:
- Add new project via Android Studio
- Add latest stable version of the necessary libraries to project module's build.gradle
- Enable Google Places API from Google Developer Console and acquire API key to be placed in the WhatPlace project so that the app can get the places data from Google
- Build the project from Android Studio to ensure build doesn't produce any errors
- Initialize git repository and commit first code

Task 2: Implement UI for Each Activity and Fragment

- Build the UI for MainActivity
- Build the UI for PlaceAddActivity (Form)
- Build the UI for CarViewFragment
- Build the UI for PlaceDetailFragment
- Build the UI for the card of a single place

Task 3: Get the adapter to display Places data no View

- Build the code for a Place detail
- Implement Google Play Service and fetch place data
- Get fetched data (place name, address) to display on a Place detail page

Task 4: User entry form

- Build the code for user entry form
- Get fetched data to autofill some parts of the form (Business name, address, website, phone)

Task 5: Build Content Provide

- Build the Content Provider
- Make data persist when user saves data on "Add Place" form
- Wire up the data to show up on the UI (cardview list shows newly saved places)

Task 6: Implement AdMod

- Integrate AdMob
- Get AdMob to show test add when viewing a detail page

Task 7: Make App Responsive

- Implement master-detail view of the app for better usage of tablet space

Task 8: Final cleanup

- Move all strings to strings.xml file and enable RTL layout switching on all layouts
- Do lint check and implement recommendations
- Test with a real user using the app, observe, and get feedback