GitHub Username: PoetryHorizon

EloteroMan

Description

Ever have a hard time finding a street corn vendor? Looking for one is harder than finding a rare Pokemon. EloteroMan solves that problem. Users are allowed to register as a Vendor or an Eater. Google maps cleanly displays a corn icon for vendors and smiley faces for eaters. Explore the taste of street corn and download today! The app will use **Java and Android Studio** to accomplish all of this! Also these will be the dependencies used and versions:

Android Studio 3.3
SDK Version 27
Gradle Version 3.3.0

Dependencies:

gms:google-services:4.20
gms.play-services-maps:16.0.0
gms.play-services-location:16.0.0
firebase:firebase-core:16.0.6
firebase:firebase-database:16.0.6
firebase:firebase-auth:16.1.0
firebase:firebase-crash:16.2.1

Plugin: com.google.gms.google-services

Intended User

This app is for users wanting to make their lives easier. Whether you are a Vendor or an Eater, using this app will eliminate the worries of looking for street corn or looking for customers.

Features

- Register/Login information stored in Firebase
- Uses Google Maps within App
- Easily differentiates Vendors from Eaters with icons displayed on Google Maps
- Widget allows users to click on widget to turn on/off GPS location

User Interface Mocks

Screen 1

Main Welcome screen. User decides if they want to register as a Vendor or an Eater of corn. When they click a button they are sent to the Login/Registration screen.

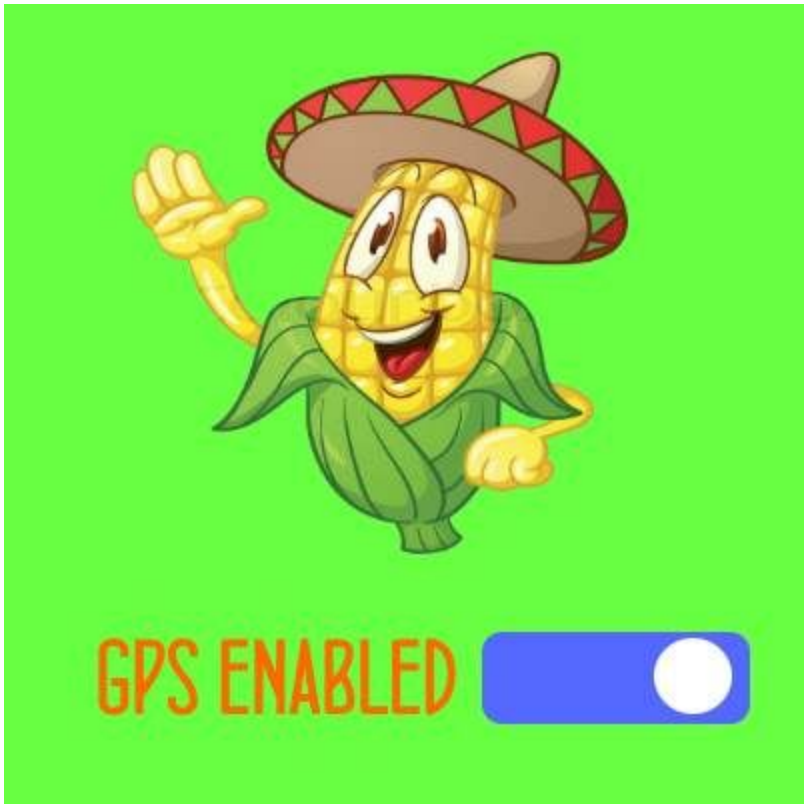
Screen 2

This is the Login/Registration Screen.

Screen 3

This is the Google Maps screen after users successfully register or login. A radius of 10 miles displays Vendors and Eaters by their icon distinction. Clicking on an icon will bring up the Google Maps app to give directions.

Screen 4



The widget will allow the user to see if GPS is turn on/off. When user clicks on widget it will send them to the App Permissions where they can choose to turn on/off the GPS tracking.

Key Considerations

How will your app handle data persistence?

The app uses Firebase Database to store users emails and passwords. This will the app to tell the difference of a Vendor and Eater based on their unique ID upon registration.

How will your app handle Strings and RTL Layout Support?

The App will store all strings in the string.xml file. The main reason for this will be to allow localization to be easily possible. Here in Los Angeles, there is a huge market for Spanish speaking Android Users. By storing all string in a simple string.xml it will make the translation process much easier and doable! RTL support? You got it! RTL support will broaden the apps usage in scenarios where users read right to left.

Describe any edge or corner cases in the UX.

Users will be able to logout using the standard burger option toolbar. When they logout they will be taken to the Main screen where they choose if they are a Vendor or an Eater again. Anytime in the app, they will be able to use the standard Android back button built into the UI to go back one step. Except, if they are in the Google Maps Activity, pressing back will exit the app but still allow them to return to the app where they left off. They must Logout to return out of the Google Maps Activity screen. That Maps Activity screen will serve as the main hub for users when using the app after they have logged in as an Eater or Vendor.

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

Picasso will be used to load the images throughout the app in order to accomplish the requirement of needing to use a 3rd party library. The reason for this is to allow for more streamline code and quicker loading times of the icons on the Maps screen.

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

The main Google Play Service being used in the app is the (1) Google Maps API. This will allow users to see other users within a certain radius of their current GPS location. Vendors will be displayed as a corn icon and eaters will be displayed as a smiley face icon. Each user can click on the opposite users icon which will bring up the intent service to open the Google Maps App and give the user directions to that person. The next Google Play Service being used will be (2) AdMob. Inherently, the app will be free first release and contain minimal Ads. In later releases, a paid version will be available, for now, only a paid version will be released. Once all bugs are squashed, user feedback is reached, and maximum ease of utility has been reached the app will be ready and deserving of a Paid release without ads.

Using AsyncTask to run my onRequestPermissionResult Method
When the User first decides if they want to be a Vendor or an Eater, they will click the chosen button. After that, onRequestPermissionResult is called which brings up a Dialog Window asking the user to allow permission of Location. Within the method, many things are happening, so making sure this runs in the background will take some stress off the main thread.

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

Download all the required images and fonts, have Firebase Database always open in a separate browser window, have a separate browser window open for Youtube to be able to quickly do needed research on any code that I may get stuck on, and have the project rubric open in a browser window at all times to compare progress.

Subtasks:

- Have Color Palette open in web browser

Task 2: Implement UI for Each Activity and Fragment

Subtasks:

- Build UI for Welcome Screen
- Build UI for Vendor Login/Registration
- Build UI for Eater Login/Registration
- Build UI for Logout Screen
- Build UI for Google Maps Activity
- Build UI Integration of Ads
- Build Widget

Task 3: Implement Firebase Database

Login into Firebase and install the needed libraries and keys to connect the app to Firebase.

Subtasks:

- Create Database Parent and children, “Vendor,” and “Eater.”

Task 4: Implement Google Maps API & AdMob

Install the required libraries into the gradle and create the Java file that connects to the UI. Once a user logins/registers they will be sent to this Java Activity via intent services. Also, install the required dependencies to allow Admob to do its job.

Subtasks:

- Assign Google Maps Icons based on whether a user is a Vendor or Eater using their unique ID’s registered with the Firebase.

Task 5: Store all strings in strings.xml and implement RTL support

Storing strings in the strings.xml will be an ongoing process as the app gets built. Strings need to be stored for easy of localization. Images will avoid using hardcoded text so when the app is translated, to Spanish for example, it will not be translatable. RTL support will be implemented by using `android:supportsRtl="true"` inside the manifest.

Task 6: Create Widget

The Widget will be created to allow the user to see if they have GPS on or off. When the user clicks on the widget it will send them to the app permissions in the phone settings and allow them to turn GPS on or off.