DFS: discounted fashion sale companion

Description

A beautifully designed application for getting information about hottest discount sale offers on your favorite fashion apparels. The application sources data from www.webhose.io to bring you the latest deals in last 24 hours. The application contains features such as mark a product as favorite, share deals with your contacts and a lot more.

Intended User

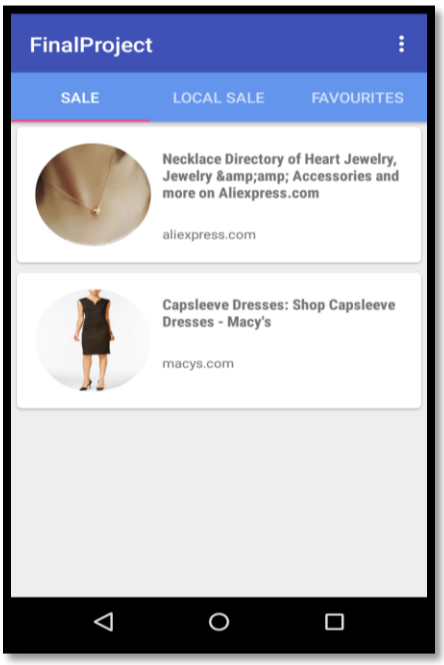
The DFS application is intended for use by the common users who are interested in fashion deals.

Features

1. Show the products
2. View product details.
3. Share the application content with other apps
4. Mark product as favourite.
5. Display products according to user's current location.

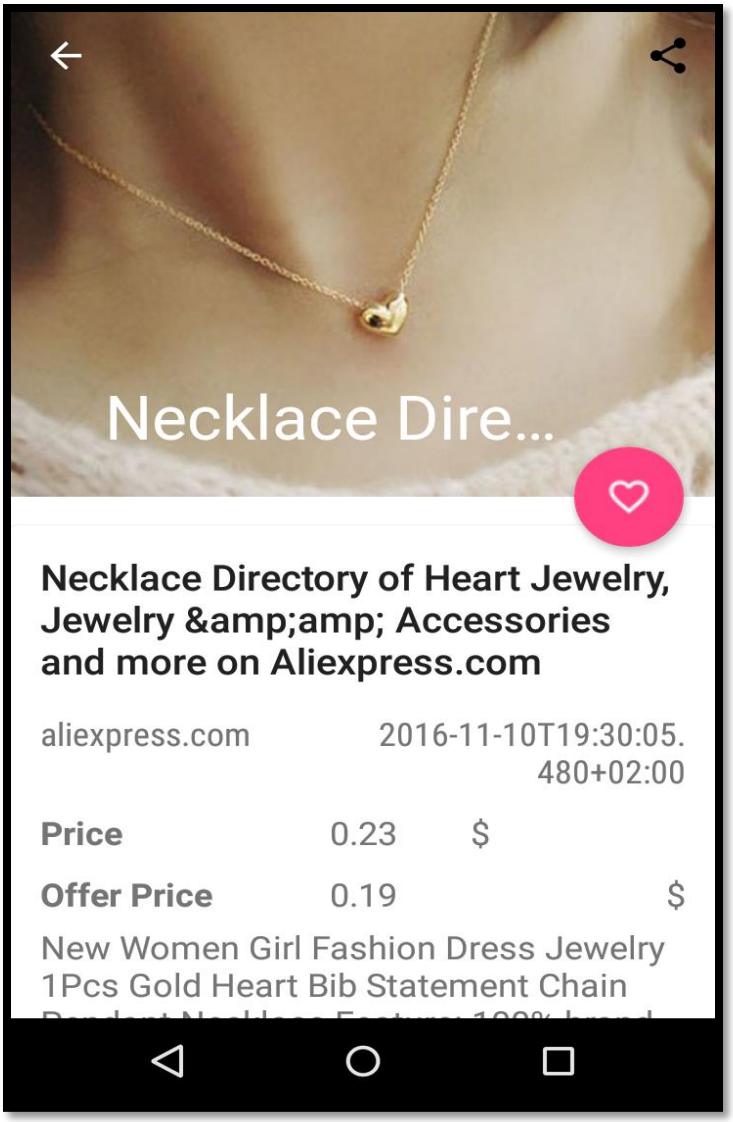
User Interface Mocks

Screen 1: Main Screen



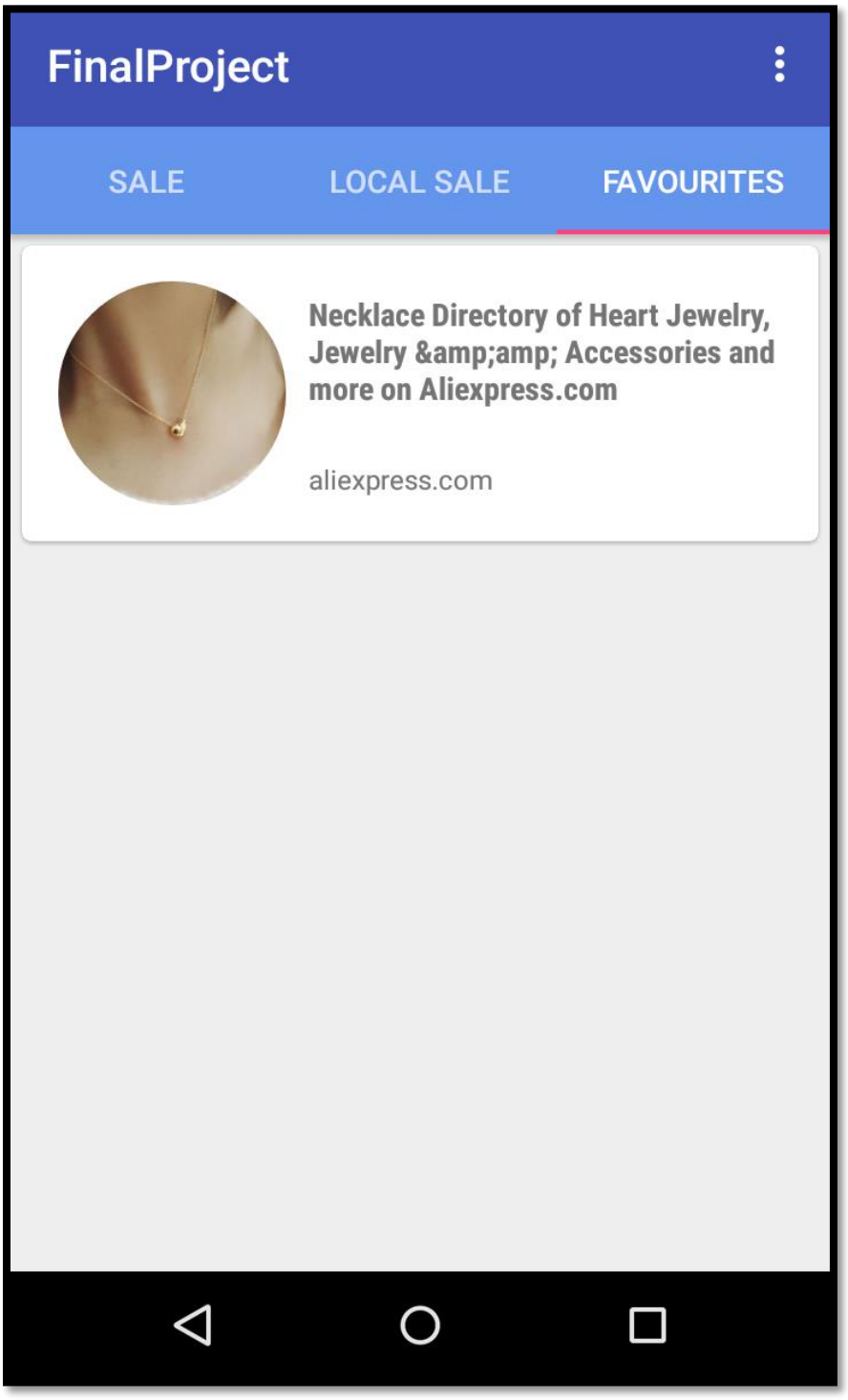
Shows list of products on online sale on the screen (retrieved from the *webhose.io api)*.

Screen 2: Product details Screen



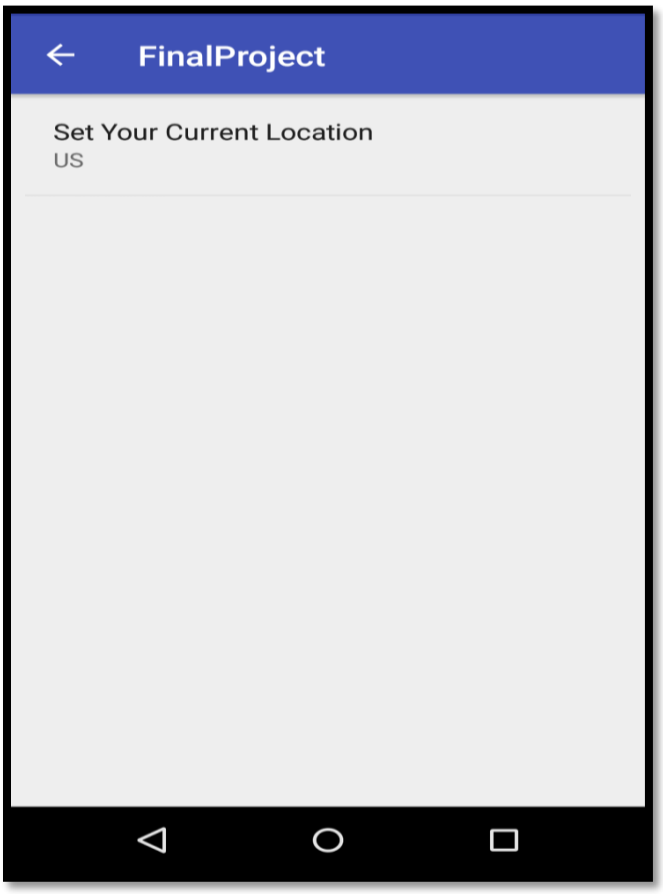
Display details of selected product.

Screen 3: Favourites Screen



Shows list of favourite products on the screen.

Screen 4: Settings Screen



To choose current country location.

Screen 5: Widget Screen

It displays the list of products.

Key Considerations

How will your app handle data persistence?

Application will use *content provider* and *shared preferences* to maintain its data locally.
DFS uses a Loader in order to move the data from the `ContentProvider` to the views

Describe any corner cases in the UX.

1. The application stores all the products information retrieved from webhose.io api in the physical storage of user's device. In case of low storage space the application might crash and/or the device may hang or slow down.
2. If the user rapidly switches between different screens of the DFS app, the app might hang.

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

Butterknife: Removing the boilerplate code like "findViewById"
Design support library: For material goodness.
Picasso : For image transformations.
CircleImageView: For image view surrounded by a circle.
RecycleView: To display data set.
Location : To find out current country location of user's device.
Analytics : To get relevant statistics about the user behavior.

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

To facilitate data transfer between the device and the server DFS application uses a *Sync Adapter* via the "Bound service".

DFS will be using *Analytics* and *location* services which are dependent on google play services.

Task 1: Project Setup

→ Import all libraries
-> Allow internet permission in manifest file.
-> Get an *api* key from webhose.io.

Task 2: Implement UI for Each Activity and Fragment

Create proper UI for the application. The following screens will be created:
1. Main screen – Activity
   a. Product Tab – Fragment
   b. Local Sale Tab - Fragment
   c. Favourite Tab – Fragment
2. Product details screen - Activity
3. Setting Screen - Activity

Task 3 : Adding data persistence.

Integrate sync adapter in the application to save the data from server in 24 hours intervals. Introduce content provider into the application to store the data locally.

Task 4 : Adding Settings options for the app

To customize user's experience the application will features a *settings* options where the user can specify default parameters e.g. current country location.