ConnectR App

1. What is RConnect?

Indian business market model is one of the most complex model. There are levels of distributors (dealers) exist before a product reaches the end customer. Our app is intended to the retail outlets. Retail outlet owners uses the app to know the list of items (products) of each category, to know the latest products available, top products of the week and can order a list of products. RConnect enhances the communication between the dealer and retail outlet owners.

RConnect, a distributor can enhance the sale through e-advertising, e commerce by replacing the traditional advertising and service management.

Animation:

Object Animator:

```
public static void animateY(ViewHolder holder,Boolean goingDown)

{

   ObjectAnimator translatorY = ObjectAnimator.ofFloat(holder.itemView, "translationY",
goingDown==true?200:-200, 0);
   translatorY.setDuration(1500);
   translatorY.start();
}
```

calling on the holder:

```
public void onBindViewHolder(ViewHolder holder, int position) {
   Map<String, ?> e_item = electronicsList.get(position);
   holder.bindeitem(e_item);
   AnimateUtils animateUtils = new AnimateUtils();
   if (position > previousPosition) {
      animateUtils.animateY(holder, true);
   } else {
      animateUtils.animateY(holder, false);
   }
   previousPosition = position;
}
```

References:

https://www.youtube.com/user/slidenerd

https://www.google.com/design/spec/material-design/introduction.html#

Data – Walmart API

Steps followed to get data from Walmart API

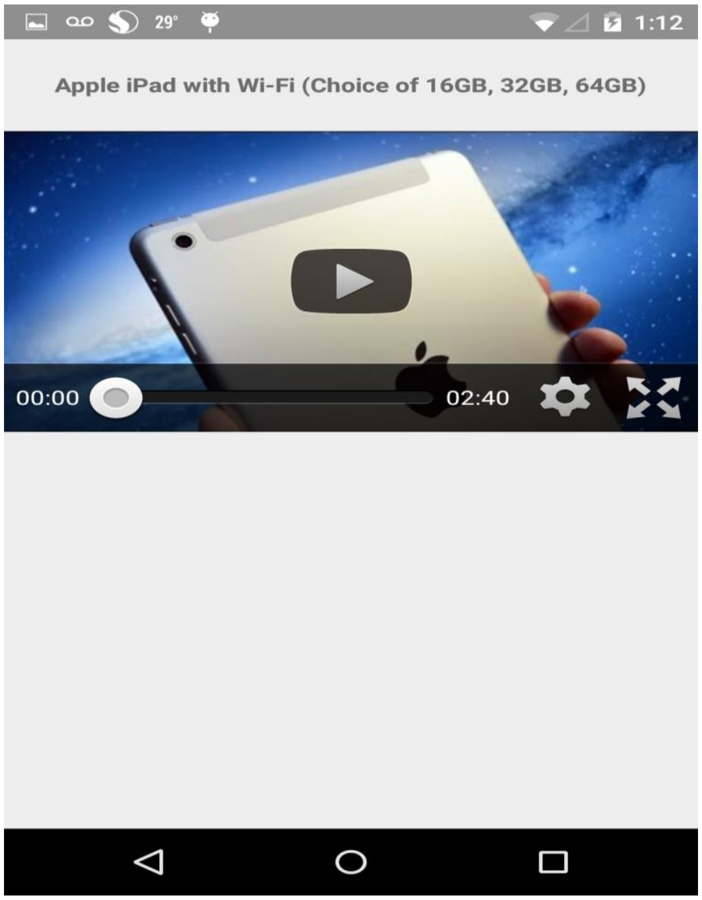
1. User should register to the developer's portal. https://developer.walmartlabs.com/member/register
2. Documentation helps to understand the API calling.
3. JSON example: http://api.walmartlabs.com/v1/paginated/items?format=json&category=3944_3951_132960&apiKey=seuyq22f8ejzzvq4fdxuh3zd
   The JSON above is for electronics category(3944395113) which in turn retrieved from another JSON

API Parsing

```
if (electronicsJson != null) {
    JSONArray electronicsJsonArray = electronicsJson.getJSONArray("items");
    if(electronicsJsonArray != null){
        for (int i = 0; i < 25; i++) {
            JSONObject eitemJSONObject = (JSONObject) electronicsJsonArray.get(i);

            if(eitemJSONObject.has("name"))
                 name = (String) eitemJSONObject.get("name");
            if(eitemJSONObject.has("salePrice") )
                 price = (Double) eitemJSONObject.get("salePrice");
            if(eitemJSONObject.has("msrp") )
                msrp = (Double) eitemJSONObject.get("msrp");
                 if(eitemJSONObject.has("brandName"))
                            brandname = (String) eitemJSONObject.get("brandName");
                       if(eitemJSONObject.has("thumbnailImage"))
                            tnailimg = (String) eitemJSONObject.get("thumbnailImage");
                       if(eitemJSONObject.has("mediumImage") )
                            mediumimg = (String) eitemJSONObject.get("mediumImage");
                       if(eitemJSONObject.has("largeImage") )
                            limg = (String) eitemJSONObject.get("largeImage");
//                        if(eitemJSONObject.has("standardShipRate") )
//                             standshiprate = (Double) eitemJSONObject.get("standardShipRate");
                       if(eitemJSONObject.has("marketplace") )
                            marketPL = (Boolean) eitemJSONObject.get("marketplace");
                       if(eitemJSONObject.has("stock"))
                            stock = (String) eitemJSONObject.get("stock");
                       if(eitemJSONObject.has("itemId"))
                           itemId = (Integer) eitemJSONObject.get("itemId");
                       if(eitemJSONObject.has("modelNumber"))
                           modelNumber = (String) eitemJSONObject.get("modelNumber");
                       if(eitemJSONObject.has("color"))
                           color = (String) eitemJSONObject.get("color");
```

4) The Parse JSON is stored in the HashMap object and fed to the respective adapters.

Using Youtube API



Steps to use Youtube API:

User should register the Android App and get the API key for making requests to a google developer console.

Credentials Page:

https://console.developers.google.com/project/gmailaccountsignon-985/apiui/credential?authuser=0

API key generated for our app: AIzaSyBN6M9QSKj8pPFpqogYkQeDd9GO65cWCW4

Download Youtube Player API

Add the jar file as lib file to the project

compile files('libs/YouTubeAndroidPlayerApi.jar')

Call Youtube activity

Activity extends YouTubeBaseActivity implements YouTubePlayer.OnInitializedListener.

Initialize Youtube Player View.

OnCreate of Youtube Activity

```
youTubeView = (YouTubePlayerView) findViewById(R.id.youtube_view);
youTubeView.initialize(YoutubeDeveloperKey, this);
youtubeString = (String) getIntent().getSerializableExtra("youtubeString");
```

YoutubePlayerView in the xml file

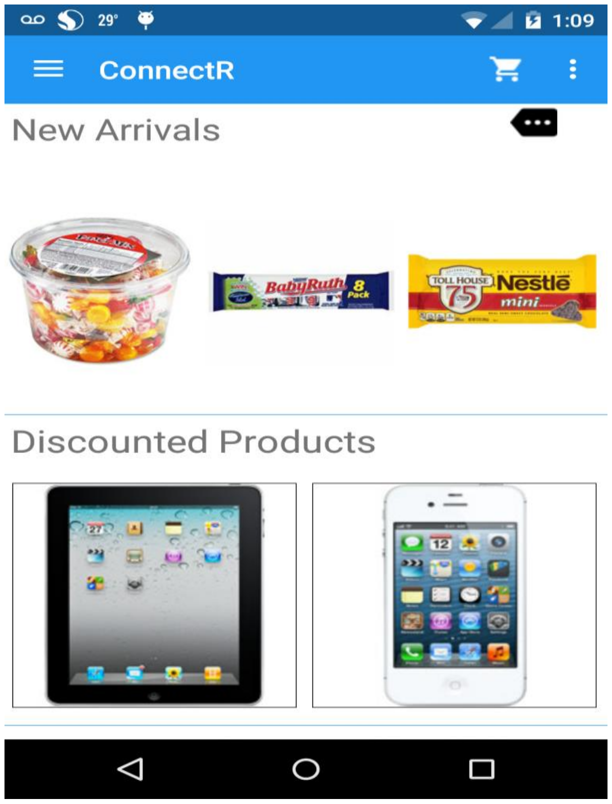
```
<com.google.android.youtube.player.YouTubePlayerView
    android:id="@+id/youtube_view"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
```

On InitializationSuccessful

```
   @Override
   public void onInitializationSuccess(YouTubePlayer.Provider provider, YouTubePlayer
youTubePlayer, boolean b) {
       YPlayer = youTubePlayer;
/*
* Now that this variable YPlayer is global you can access it
* throughout the activity, and perform all the player actions like
* play, pause and seeking to a position by code.
*/
       if (!b) {
           //YPlayer.cueVideo("KQ4br64b7Mc");
           YPlayer.cueVideo(youtubeString);
       }
   }
```

Reference: http://javatechig.com/android/youtubeplayerview-example-in-android-using-youtube-api

ViewFlipper



XML file

```
<LinearLayout
   android:layout_below="@id/ll1"
   android:layout_width="match_parent"
   android:orientation="horizontal"
   android:layout_height="200dp"

   android:id="@+id/rel1"
   >

   <ViewFlipper
      android:id="@+id/viewFlipper1"
      android:layout_width="fill_parent"
      android:layout_weight="1"
      android:padding="1dp"
      android:layout_height="match_parent">
   </ViewFlipper>
   <ViewFlipper
      android:id="@+id/viewFlipper2"
      android:layout_weight="1"

      android:padding="1dp"
      android:layout_width="fill_parent"
      android:layout_height="match_parent">
   </ViewFlipper>
```

```
    <ViewFlipper
        android:id="@+id/viewFlipper3"
        android:layout_weight="1"
        android:padding="1dp"
        android:layout_width="fill_parent"
        android:layout_height="match_parent">
    </ViewFlipper>
</LinearLayout>
```

onCreateView

```
while (i < mvData.getSize() - 3) {
```

Defining the layout for the flippers

```
    LinearLayout.LayoutParams flipper_layout_params = new
LinearLayout.LayoutParams(250, 600);
    LinearLayout flipper1ImgesLayout = new LinearLayout(getActivity());
    flipper1ImgesLayout.setOrientation(LinearLayout.HORIZONTAL);
    flipper1ImgesLayout.setLayoutParams(flipper_layout_params);

    LinearLayout flipper2ImgesLayout = new LinearLayout(getActivity());
    flipper2ImgesLayout.setOrientation(LinearLayout.HORIZONTAL);
    flipper2ImgesLayout.setLayoutParams(flipper_layout_params);

    LinearLayout flipper3ImgesLayout = new LinearLayout(getActivity());
    flipper3ImgesLayout.setOrientation(LinearLayout.HORIZONTAL);
    flipper3ImgesLayout.setLayoutParams(flipper_layout_params);
```

Programatically creating the imageview for the layouts.

```
    ImageView img1 = new ImageView(getActivity());
    ImageView img2 = new ImageView(getActivity());
    ImageView img3 = new ImageView(getActivity());
    LinearLayout.LayoutParams imgparams = new LinearLayout.LayoutParams(240, 400);
    img1.setLayoutParams(imgparams);
    img2.setLayoutParams(imgparams);
    img3.setLayoutParams(imgparams);
    //img1.setImageResource((Integer) mvData.getItem(i).get("largeImage"));
```

Async download of the images

```
    String url = (String) mvData.getItem(i).get("largeImage");
    MyDownloadImageAsyncTask task = new MyDownloadImageAsyncTask(img1);
    task.execute(new String[]{url});
   // img2.setImageResource((Integer) mvData.getItem(i + 1).get("largeImage"));
   // img3.setImageResource((Integer) mvData.getItem(i + 2).get("largeImage"));
```

Assigning the imageviews to the flippers respectively

```
    flipper1ImgesLayout.addView(img1);
    flipper2ImgesLayout.addView(img2);
    flipper3ImgesLayout.addView(img3);
    mViewFlipper1.addView(flipper1ImgesLayout);
    mViewFlipper2.addView(flipper2ImgesLayout);
    mViewFlipper3.addView(flipper3ImgesLayout);

    i = i + 1;
}
```

Set the interval and auto start of the flipper(s)

```
mViewFlipper1.setAutoStart(true);
mViewFlipper1.setFlipInterval(3000);
mViewFlipper1.startFlipping();
```

Set the animation for the viewflipper

```
mViewFlipper1.setInAnimation(getActivity(), R.animator.in_from_right);
mViewFlipper1.setOutAnimation(getActivity(), R.animator.out_to_left);
```

Animator- out to left

```
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:shareInterpolator="false">
    <translate android:fromXDelta="0%" android:toXDelta="-100%"
        android:fromYDelta="0%" android:toYDelta="0%"
        android:duration="500"/>
</set>
```

Database

We use Parse.com cloud to register an user, authenticate and to store other information related to our app.

What is Parse.com?

Parse is a cloud to power your android app. It provides complete backend solutions to the app. Best part of the Parse: It makes the app to have minimum configuration to work.

How Parse cloud is used.

Register the app and use the application id and client key(which you get when one register to Parse.com)

Advantages:

It provides framework for user sign up and login for an app.

Any kind of data related to app can be stored in the cloud as objects of Parse as ParseObject.

Data retrieve is easy

It provides data security

It provides a framework to have a local datastore which can be created with an app, local to app and later sync to the cloud.

Querying the Parse.com datastore is easy

Steps to use Parse.com cloud with local datastore

1. Setting the permissions .

```
<permission
    android:name="com.parse.starter.permission.C2D_MESSAGE"
    android:protectionLevel="signature" />

<uses-permission android:name="com.parse.starter.permission.C2D_MESSAGE" />
```

2. Add the parse API key and start the initialization

```
Parse.initialize(this, "Ffwykq9dpASYCdQf46kSABr6HaNc9FkowWxbcpa5",
"AhIe9wFMdII2jWFNivcCSArdLg2KRgj57Rzj7LTa");
ParseInstallation.getCurrentInstallation().saveInBackground();
```

3. Add the Jar files to the lib folder and add the files under the dependencies section

```
compile fileTree(dir: 'libs', include: 'Parse-*.jar')
compile fileTree(dir: 'libs', include: 'ParseCrashReporting-*.jar')
```

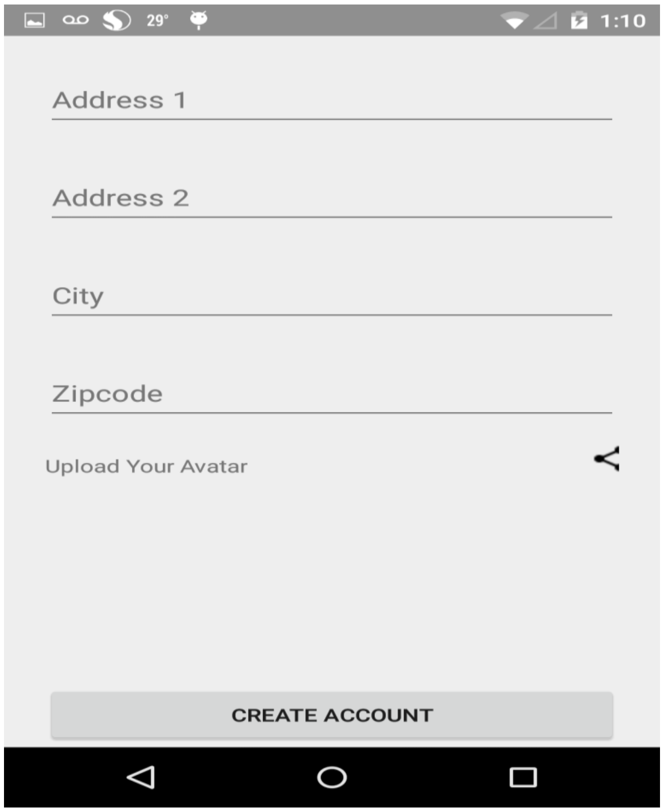
4. Enable the local data store

```
// Enable Local Datastore.
Parse.enableLocalDatastore(this);

//Register the class- Data_CartObject
ParseObject.registerSubclass(Data_CartObject.class);
```

5. To use anonymous user when an user is not logged in

```
// Enable Local Datastore.
Parse.enableLocalDatastore(this);

//Register the class- Data_CartObject
ParseObject.registerSubclass(Data_CartObject.class);
```

How Sign Up works

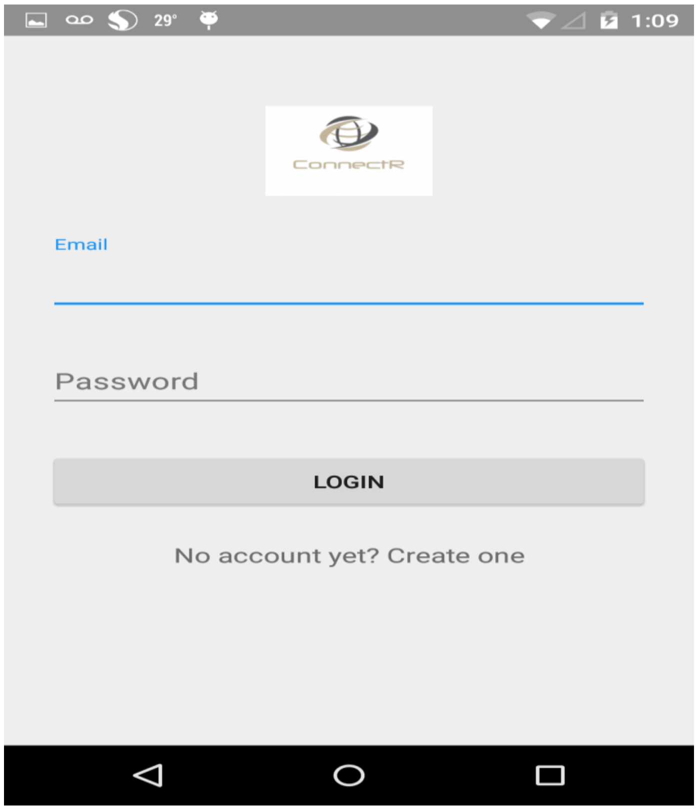


```java
public void signup() {
    Log.d(TAG, "Signup");

    if (!validate()) {
        onSignupFailed();
        return;
    }

    _signupButton.setEnabled(false);

    final ProgressDialog progressDialog = new ProgressDialog(Activity_Signup.this,
            R.style.AppTheme_Dark_Dialog);
    progressDialog.setIndeterminate(true);
    progressDialog.setMessage("Creating Account...");
    progressDialog.show();

    String name = _nameText.getText().toString();
    String email = _emailText.getText().toString().trim();
    String password = _passwordText.getText().toString().trim();
    String address1 = _address1.getText().toString();
    String address2 = _address2.getText().toString();
    String city = _city.getText().toString();
    String zip = _zip.getText().toString();


    // Get the user information to save to cloud
```

```
ParseUser user = new ParseUser();
user.setUsername(email);
user.setPassword(password);
user.put("name", name);
user.put("address1", address1);
user.put("address2",address2 );
user.put("city", city);
user.put("zip", zip);
if(profileImgString!= null)
user.put("profileImageString", profileImgString);

// Call the Parse signup method
user.signUpInBackground(new SignUpCallback() {
    @Override
    public void done(ParseException e) {
      // dialog.dismiss();
        if (e != null) {
            // Show the error message
            onSignupFailed();
        } else {
            // Start an intent for the dispatch activity
            onSignupSuccess();

        }
    }
});
```

Reference Link: https://parse.com/tutorials/using-the-local-datastore

How Login works:



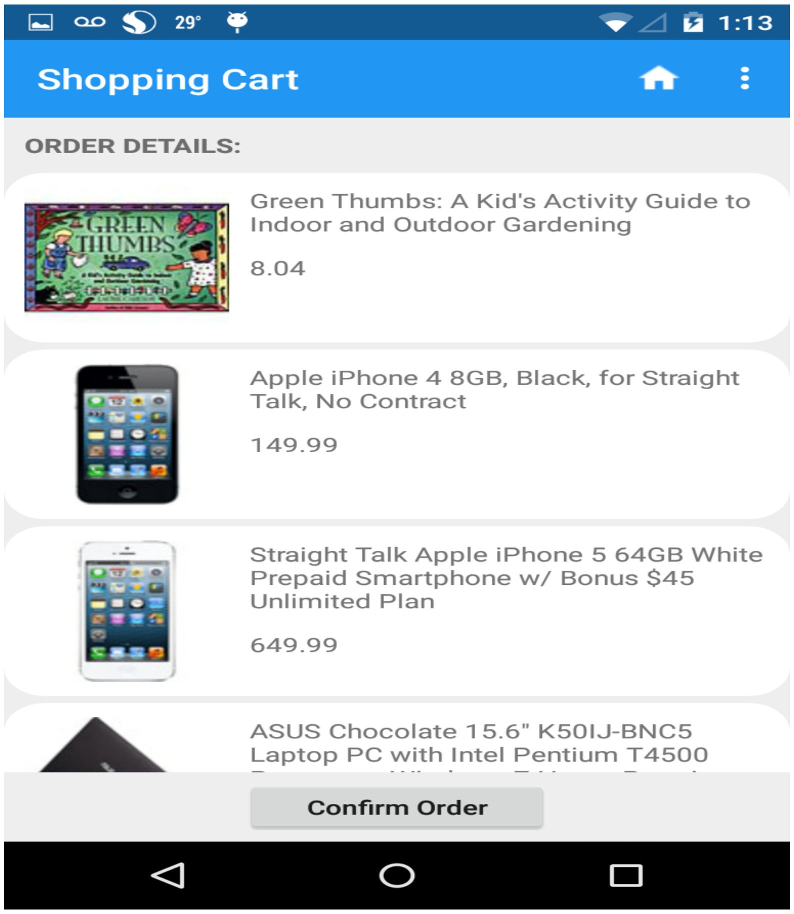
//Retrieve the email id and password entered by user

```
String email = _emailText.getText().toString().trim();
String password = _passwordText.getText().toString().trim();
```

```
// call the loginINbackground ParseUser function


ParseUser.logInInBackground(email, password, new LogInCallback() {
    @Override
    public void done(ParseUser user, ParseException e) {
        if (e != null) {
            // Show the error message

            onLoginFailed();
        } else {

            if (user.get("profileImageString") != null) {
                prof_img_string = (String) user.get("profileImageString");
            }


            Toast.makeText(getBaseContext(), "Login Successful",
Toast.LENGTH_LONG).show();
```

```
            onLoginSuccess();
        }
    }
});
```

How data is saved to cloud



Green Thumbs: A Kid's Activity Guide to Indoor and Outdoor Gardening
8.04

Apple iPhone 4 8GB, Black, for Straight Talk, No Contract
149.99

Straight Talk Apple iPhone 5 64GB White Prepaid Smartphone w/ Bonus $45 Unlimited Plan
649.99

ASUS Chocolate 15.6" K50IJ-BNC5 Laptop PC with Intel Pentium T4500

1. In our app, we store the items that are added to the cart to cloud. In future if user wants to know the history of his shopping, the details are rendered to user on the app with a call to cloud.

2. The data is stored in the form of ParseObjects. So to save the cart items to the cloud, we created a class that extends the ParseObject Class

   Code snippet:

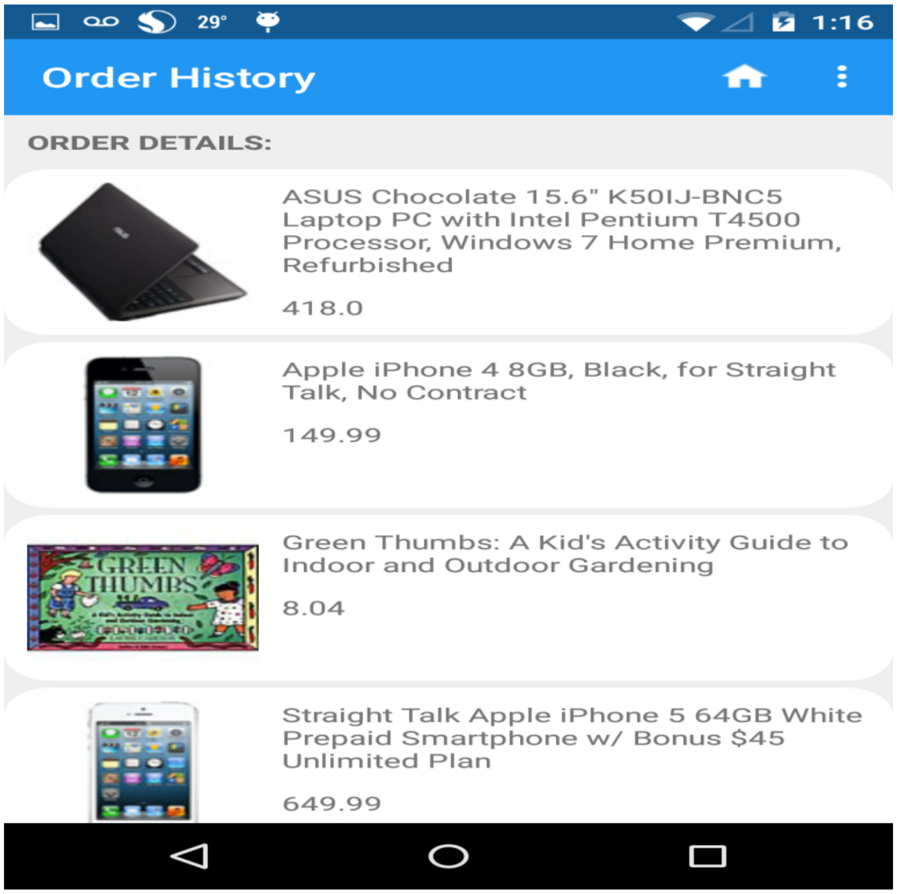
```
@ParseClassName("Data_CartObject")
public class Data_CartObject extends ParseObject {

    public String getImageURL(){
        return getString("imageUrl");
    }
```

```
public void setImageURL(String imgUrl){
    put("imageUrl", imgUrl);
}

public String getProductName(){
    return getString("productname");
}

public void setProductName(String prdName){
    put("productname", prdName);
}

public Double getUnitCost(){
    return getDouble("unitCost");
}

public void setUnitCost(Double unitCost){
    put("unitCost", unitCost);
}
```

3. We have a local datastore to save the data when an item is added to the cart.

```
item_cart.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        item_cart.setBackgroundResource(android.R.color.black);
        if (electronic_item != null) {
            ParseUser currUser;
            Data_CartObject cart_item = new Data_CartObject();
            cart_item.setProductName((String) electronic_item.get("name"));
            cart_item.setUnitCost((Double) electronic_item.get("salePrice"));
            cart_item.setImageURL((String) electronic_item.get("thumbnailImage"));
            currUser = ParseUser.getCurrentUser();
            cart_item.setOwner(currUser);
            cart_item.setDraft(true);
            cart_item.pinInBackground();
        }
```

4. When user confirms the order, the items are stored to the cloud



Order History

ORDER DETAILS:

ASUS Chocolate 15.6" K50IJ-BNC5 Laptop PC with Intel Pentium T4500 Processor, Windows 7 Home Premium, Refurbished

418.0

Apple iPhone 4 8GB, Black, for Straight Talk, No Contract

149.99

Green Thumbs: A Kid's Activity Guide to Indoor and Outdoor Gardening

8.04

Straight Talk Apple iPhone 5 64GB White Prepaid Smartphone w/ Bonus $45 Unlimited Plan

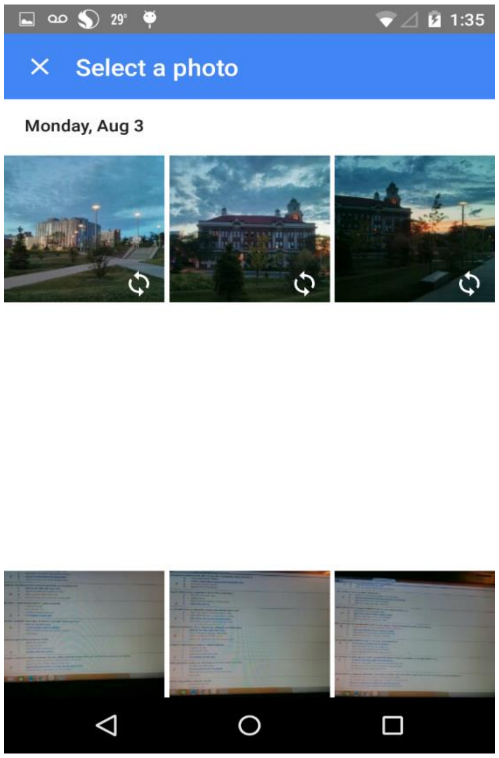
649.99

```
private void syncCartToParse(){

    // Check the connection exists to the Parse

    ConnectivityManager cm = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo ni = cm.getActiveNetworkInfo();

    if((ni != null) && (ni.isConnected())){
        if(!ParseAnonymousUtils.isLinked(ParseUser.getCurrentUser())){
            // If we have a network connection and a current logged in user,
            // sync the cart to the cloud

            // In this app, local changes should overwrite content on the
            // server.

            ParseQuery<Data_CartObject> query = Data_CartObject.getQuery();
            query.fromPin();
            query.whereEqualTo("isDraft", true);
            query.findInBackground(new FindCallback<Data_CartObject>() {
                @Override
                public void done(List<Data_CartObject> list, ParseException e) {
                    if(e == null){
                        for(final Data_CartObject cart_obj : list){
                            cart_obj.setDraft(false);
                            cart_obj.setOwner(ParseUser.getCurrentUser());
                            cart_obj.saveInBackground(new SaveCallback() {
                                @Override
                                public void done(ParseException e) {
                                    if(e == null){
                                        // Let adapter know to update view
                                        if(!isFinishing()){
                                            cartAdapter.clear();
                                            cartAdapter.notifyDataSetChanged();
                                        }
                                    }
                                    else
                                    {
                                        cart_obj.setDraft(true);
                                    }
                                }
                            });
                        }
                    }else{
                        Log.i("CartActivity", "syncTodosToParse: Error finding
pinned" + e.getMessage());
                    }
                }
            });
```

5. As the local datastore get synced to the cloud datastore, we can use the similar logic above to get the desired data.

Reference links:
https://parse.com/docs/android/api/com/parse/ParseQuery.html
https://parse.com/docs/android/api/com/parse/ParseObject.html
https://parse.com/docs/android/guide
https://parse.com/tutorials/using-the-local-datastore

Gallery Integration



Steps:

1. Activity of the Gallery is called

```
                Intent i = new Intent(Intent.ACTION_PICK,
  MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
  startActivityForResult(i, RESULT_LOAD_IMG);
```

2. Select the Image from gallery which results back the activity which called the Gallery activity.

3. Image bitmap is retrieved as below

```
  @Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
   super.onActivityResult(requestCode, resultCode, data);

   if(requestCode == RESULT_LOAD_IMG && resultCode == this.RESULT_OK &&
data != null){


  // Get the imagepath
      Uri selectedImage = data.getData();
      String[] filePathColumn = { MediaStore.Images.Media.DATA };
```

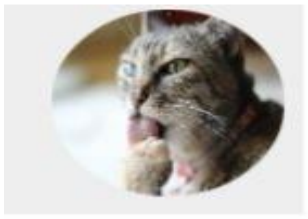
```
            Cursor cursor = this.getContentResolver().query(selectedImage,
                    filePathColumn, null, null, null);
            cursor.moveToFirst();

            int columnIndex = cursor.getColumnIndex(filePathColumn[0]);
            String picturePath = cursor.getString(columnIndex);
            cursor.close();


        //Convert the file to bitmap since it is a image
            Bitmap bitmap = BitmapFactory.decodeFile(picturePath);
            bitmap = bitmap.createScaledBitmap(bitmap, 100, 100, true);

            _display.setImageBitmap(ImageHelper.getRoundedCornerBitmap(bitmap,10));

            //Convert BitMap to string to save in database

            ByteArrayOutputStream stream = new ByteArrayOutputStream();
            bitmap.compress(Bitmap.CompressFormat.PNG, 70, stream);
            byte [] byte_arr = stream.toByteArray();
            profileImgString = Base64.encodeToString(byte_arr, Base64.DEFAULT);


        }
```

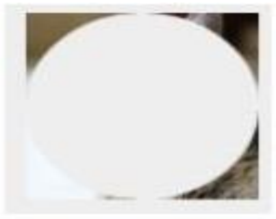
Circling an image

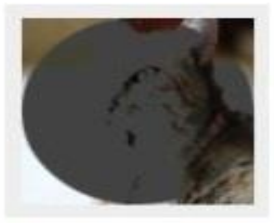
User provides the bitmap and rectangle parameters which gives the shape to the canvas.
For the given dimension the image is cropped.
What type of source that has to be in the canvas is selected, here we select PotterDuff.Mode.SRC_IN.

```
public static Bitmap getRoundedCornerBitmap(Bitmap bitmap, int pixels) {
    Bitmap output = Bitmap.createBitmap(bitmap.getWidth(), bitmap
            .getHeight(), Bitmap.Config.ARGB_8888);
    Canvas canvas = new Canvas(output);

    final int color = 0xff424242;
    final Paint paint = new Paint();
    final Rect rect = new Rect(0, 0, bitmap.getWidth(), bitmap.getHeight());
    final RectF rectF = new RectF(rect);
    final float roundPx = pixels;

    paint.setAntiAlias(true);
    canvas.drawARGB(0, 0, 0, 0);
    paint.setColor(color);
    canvas.drawRoundRect(rectF, roundPx, roundPx, paint);

    paint.setXfermode(new PorterDuffXfermode(PorterDuff.Mode.SRC_IN));
    canvas.drawBitmap(bitmap, rect, rect, paint);
    return output;
}
```

Different ways of mode selection


SRC_IN


Outer


Darker

Reference: http://stackoverflow.com/questions/11337679/porterduffxfermode-clear-a-section-of-a-bitmap