Project Code

Inventory App

Student: Keith Mullins

Supervisor: Nigel Whyte

Student ID: C00118202

Contents

Android App

AddItemActivity.java

```
package com.example.myfirstapp;

import java.io.ByteArrayOutputStream;
import java.io.InputStream;
import java.net.URL;
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.entity.BufferedHttpEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import com.parse.GetCallback;
import com.parse.ParseException;
import com.parse.ParseFile;
import com.parse.ParseObject;
import com.parse.ParseQuery;
import com.parse.ParseUser;
import com.parse.SaveCallback;
import com.squareup.picasso.Picasso;
import android.app.AlertDialog;
import android.content.Context;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.MotionEvent;
import android.view.View;
import android.view.Window;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.Toast;

public class AddItemActivity extends BaseActivity  {

    // declare variables
    byte[] byteArray;
    Bitmap bp;
    boolean notPhoto = false;

    private ParseFile imageFile;
    private ItemDetails item;

    private EditText titleEditText;
    private EditText detailsEditText;
    private EditText categoryEditText;
    private EditText authorEditText;
    private EditText publishEditText;

    private String itemTitle;
    private String itemDetails;
```

```
    private String itemCategory;
    private String itemImage;
    private String itemAuthor_artist;
    private String barcode;
    private String itemReleaseDate;

    private ImageView imgView;

    // edit text focus storage
    private String evalue;
    private String ocrText;

    private Button saveNoteButton;
    private Button cancelButton;
    private ImageButton photoButton;
    private Button ocrButton;

    Context context = this;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        // progress spinner
        requestWindowFeature(Window.FEATURE_INDETERMINATE_PROGRESS);
        // thread
        new Thread(new Runnable() {
            public void run(){
               //All your heavy stuff here!!!
            }
        }).start();


        super.onCreate(savedInstanceState);

        // set the background to empty
        LinearLayout  linearLayout = (LinearLayout)
findViewById(R.id.linearLayoutid);
        linearLayout.setBackgroundResource(0);

        // layout for list view
        LayoutInflater inflater = (LayoutInflater) this
                .getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        View contentView = inflater.inflate(R.layout.add_item, null,
false);
        mDrawerLayout.addView(contentView, 0);

        // create an intent
        Intent intent = this.getIntent();


        authorEditText = (EditText) findViewById(R.id.author_artist);
        titleEditText = (EditText) findViewById(R.id.itemTitle);
        detailsEditText = (EditText) findViewById(R.id.itemDetailsInput);
        categoryEditText = (EditText) findViewById(R.id.itemCategory);
        publishEditText = (EditText)
findViewById(R.id.publish_release_date);

        // setting imgview details
        imgView = (ImageView) findViewById(R.id.imageView1);
        int width = 260;
        int height = 326;
```

```
        LinearLayout.LayoutParams parms = new
LinearLayout.LayoutParams(width,height);
        imgView.setLayoutParams(parms);

        // if received intent from another activity
        if(intent !=null)
        {
            // which activity
            String strdata = intent.getExtras().getString("Uniqid");
            if(strdata.equals("from_Main"))
            {
                if (intent.getExtras() != null) {
                    // item not found
                    if(intent.getExtras().containsKey("notFound")){
                        Toast.makeText(getApplicationContext(), "Item not
found. Please enter it manually", Toast.LENGTH_SHORT).show();
                    }
                    else{
                    // get item details
                    String title = intent.getStringExtra("title");
                    String descritpion =
intent.getStringExtra("description");
                    String category = intent.getStringExtra("category");
                    String image = intent.getStringExtra("image");
                    String author = intent.getStringExtra("author");
                    String date = intent.getStringExtra("release");

                    // set details
                    titleEditText.setText(title);
                    detailsEditText.setText(descritpion);
                    categoryEditText.setText(category);
                    authorEditText.setText(author);
                    publishEditText.setText(date);
                    itemImage = image;

                    // set the image into a ParseFile and set imgview
                    try{
                        URL url = new URL(image);
                        HttpGet httpRequest = null;
                        httpRequest = new HttpGet(url.toURI());

                        HttpClient httpclient = new DefaultHttpClient();
                        HttpResponse response = (HttpResponse) httpclient
                                .execute(httpRequest);

                        HttpEntity entity = response.getEntity();
                        BufferedHttpEntity b_entity = new
BufferedHttpEntity(entity);
                        InputStream input = b_entity.getContent();

                        Bitmap bitmap = BitmapFactory.decodeStream(input);

                        ByteArrayOutputStream stream = new
ByteArrayOutputStream();
                        bitmap.compress(Bitmap.CompressFormat.PNG, 100,
stream);
                        byteArray = stream.toByteArray();
                        imageFile = new ParseFile("photo.jpg",byteArray);

                        imgView.setImageBitmap(bitmap);
```

```
                    } catch (Exception ex){
                        // Error
                    }
                }
              }
            }
            if(strdata.equals("from_OCR"))
            {
                // From Action settings Menu
                // Do Nothing

            }
            // if item was click on in the list
            if(strdata.equals("from_List_Activity"))
            {
                if (intent.getExtras() != null) {
                    // get item details
                    item = new ItemDetails(intent.getStringExtra("itemId"),
intent.getStringExtra("itemTitle"), intent.getStringExtra("itemContent"),
intent.getStringExtra("itemCategory"),
intent.getStringExtra("itemAuthor_artist"),
intent.getStringExtra("itemReleaseDate"));
                    String imageUrl = intent.getStringExtra("itemImage");
                    titleEditText.setText(item.getTitle());
                    detailsEditText.setText(item.getContent());
                    categoryEditText.setText(item.getCategory());
                    authorEditText.setText(item.getAuthor_Artist());
                    publishEditText.setText(item.getReleaseDate());
                    // load img view from url
                    Picasso.with(this).load(imageUrl).into(imgView);
                }
            }
            if(strdata.equals("from_List_Activity_Menu"))
            {
                // From Action settings Menu
                // Do Nothing

            }
        }
        // cancel button
        cancelButton = (Button)findViewById(R.id.cancel_button);
        cancelButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
        // save item button
        saveNoteButton = (Button)findViewById(R.id.save_button);
        saveNoteButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                saveItem();
            }
        });
        // open camera button
        photoButton = (ImageButton)findViewById(R.id.camera_button);
        photoButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                openCamera();
```

```
            }
        });

        // which textbox was clicked
        titleEditText.setOnTouchListener(new View.OnTouchListener() {
            @Override
            public boolean onTouch(View arg0, MotionEvent arg1) {
                evalue = "1";
                return false;
            }
        });
        detailsEditText.setOnTouchListener(new View.OnTouchListener() {

            @Override
            public boolean onTouch(View arg0, MotionEvent arg1) {
                evalue = "2";
                return false;
            }
        });

        // put ocr text into whichever text box clicked on
        ocrButton = (Button)findViewById(R.id.ocr_button);
        ocrButton.setOnClickListener(new View.OnClickListener()
        {
            public void onClick(View arg0) {
                if(evalue=="1")
                {
                    Intent intent = new Intent(context,
com.example.myfirstapp.camerabase.CaptureActivity.class);
                    startActivityForResult(intent, 1);

                }
                else if(evalue=="2")
                {
                    Intent intent = new Intent(context,
com.example.myfirstapp.camerabase.CaptureActivity.class);
                    startActivityForResult(intent, 2);
                }
                else{
                    Toast.makeText(getApplicationContext(), "Please select
a text box first then click the ocr button", Toast.LENGTH_SHORT).show();
                }
            }
        });
    }

    // save the item to the cloud
    private void saveItem() {

        itemTitle = titleEditText.getText().toString();
        itemDetails = detailsEditText.getText().toString();
        itemCategory = categoryEditText.getText().toString();
        itemAuthor_artist = authorEditText.getText().toString();
        itemReleaseDate = publishEditText.getText().toString();

        itemTitle = itemTitle.trim();
        itemDetails = itemDetails.trim();
        itemCategory = itemCategory.trim();
        itemAuthor_artist = itemAuthor_artist.trim();
        itemReleaseDate = itemReleaseDate.trim();
```

```
        // If user doesn't enter a title or content, do nothing
        // If user enters title, but no content, save
        // If user enters content with no title, give warning
        // If user enters both title and content, save

        if (!itemTitle.isEmpty() && !itemCategory.isEmpty()) {

            // Check if item is being created or edited
             if (item == null) {
                // create new item

                ParseObject post = new ParseObject("Inventory");
                itemTitle = itemTitle.toLowerCase();
                itemDetails = itemDetails.toLowerCase();
                itemCategory = itemCategory.toLowerCase();
                itemAuthor_artist = itemAuthor_artist.toLowerCase();
                itemReleaseDate = itemReleaseDate.toLowerCase();

                // save item details
                post.put("title", itemTitle);
                post.put("description", itemDetails);
                post.put("category", itemCategory);
                post.put("author_artist", itemAuthor_artist);
                post.put("releaseDate", itemReleaseDate);
                post.put("barcode", "0");
                if(!notPhoto){
                    // if try to save item with no image - give defualt
image
                    if(imageFile == null){
                        bp = BitmapFactory.decodeResource(getResources(),
R.drawable.no_image);
                        ByteArrayOutputStream stream = new
ByteArrayOutputStream();
                        bp.compress(Bitmap.CompressFormat.PNG, 100,
stream);
                        byteArray = stream.toByteArray();
                        imageFile = new ParseFile("photo.jpg",byteArray);
                        itemImage = "blank";
                    }
                    post.put("image", itemImage);
                    post.put("photo", imageFile);
                }
                // else put blank image
                else{
                    ByteArrayOutputStream stream = new
ByteArrayOutputStream();
                    bp.compress(Bitmap.CompressFormat.PNG, 100, stream);
                    byteArray = stream.toByteArray();
                    imageFile = new ParseFile("photo.jpg",byteArray);
                    post.put("photo", imageFile);
                    notPhoto = false;
                }
                // save for current user
                post.put("author", ParseUser.getCurrentUser());
                setProgressBarIndeterminateVisibility(true);
                post.saveInBackground(new SaveCallback() {
                    public void done(ParseException e) {
                        setProgressBarIndeterminateVisibility(false);
                        if (e == null) {
                            // Saved successfully.
```

```
                            Toast.makeText(getApplicationContext(),
"Saved", Toast.LENGTH_SHORT).show();
                            finish();
                        } else {
                            // The save failed.
                            Toast.makeText(getApplicationContext(), "Failed
to Save", Toast.LENGTH_SHORT).show();
                            Log.d(getClass().getSimpleName(), "User update
error: " + e);
                        }
                    }
                });

            }
            else {
                // update item
                // select from database
                ParseQuery<ParseObject> query =
ParseQuery.getQuery("Inventory");

                // Retrieve the object by id
                query.getInBackground(item.getId(), new
GetCallback<ParseObject>() {
                  public void done(ParseObject post, ParseException e) {
                    if (e == null) {
                      // Now let's update it with some new data.
                        itemTitle = itemTitle.toLowerCase();
                        itemDetails = itemDetails.toLowerCase();
                        itemCategory = itemCategory.toLowerCase();
                        itemAuthor_artist =
itemAuthor_artist.toLowerCase();
                        itemReleaseDate = itemReleaseDate.toLowerCase();
                        post.put("title", itemTitle);
                        post.put("description", itemDetails);
                        post.put("category", itemCategory);
                        post.put("author_artist", itemAuthor_artist);
                        post.put("releaseDate", itemReleaseDate);
                        post.put("barcode", "0");
                        if(notPhoto){
                            ByteArrayOutputStream stream = new
ByteArrayOutputStream();
                            bp.compress(Bitmap.CompressFormat.PNG, 100,
stream);
                            byteArray = stream.toByteArray();
                            imageFile = new
ParseFile("photo.jpg",byteArray);
                            post.put("photo", imageFile);
                            notPhoto = false;
                        }

                        post.put("author", ParseUser.getCurrentUser());
                        setProgressBarIndeterminateVisibility(true);
                        post.saveInBackground(new SaveCallback() {
                            public void done(ParseException e) {

setProgressBarIndeterminateVisibility(false);
                                if (e == null) {
                                    // Saved successfully.
                                    Toast.makeText(getApplicationContext(),
"Saved", Toast.LENGTH_SHORT).show();
```

```
                                    Intent intent = new
Intent(getApplicationContext(), ListItemActivity.class);
                                    //startActivity(intent);
                                    finish();
                                } else {
                                    // The save failed.
                                    Toast.makeText(getApplicationContext(),
"Failed to Save", Toast.LENGTH_SHORT).show();
                                    Log.d(getClass().getSimpleName(), "User
update error: " + e);
                                }
                            }
                        });
                    }
                }
            });
        }
    }
    // validation for user not entering data and pressing save
    else if (itemTitle.isEmpty() || itemCategory.isEmpty()) {
        String alertHeader = "Please Enter: ";
        String alertTitle = "Title";
        String alertCate = "Category";
        AlertDialog.Builder builder = new
AlertDialog.Builder(AddItemActivity.this);
        builder.setMessage(alertHeader + "\n" + alertTitle + "\n" +
alertCate)
            .setTitle(R.string.edit_error_title)
            .setPositiveButton(android.R.string.ok, null);
        AlertDialog dialog = builder.create();
        dialog.show();
    }
}

public void openCamera(){
      Intent intent = new
Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);
      startActivityForResult(intent, 0);
}
    // check which activity returned a result and get the data
    protected void onActivityResult(int requestCode, int resultCode,
Intent data){
        switch(requestCode){
            case 0:
                if(resultCode != RESULT_CANCELED){
              super.onActivityResult(requestCode, resultCode, data);
              notPhoto = true;
              bp = (Bitmap) data.getExtras().get("data");
              imgView.setImageBitmap(bp);
                }
            break;
            case 1:
                if(data!=null){
                    ocrText = data.getStringExtra("ocr");
                    titleEditText.setText(ocrText);
                }
            break;
            case 2:
                if(data != null){
                ocrText = data.getStringExtra("ocr");
                detailsEditText.setText(ocrText);
```

```
                }
            break;
        }
    }
}
```

BaseActivity.java

```
package com.example.myfirstapp;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.StringReader;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLConnection;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.util.HashMap;
import java.util.Map;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import org.jsoup.Jsoup;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.InputSource;
import org.xml.sax.SAXException;
import jim.h.common.android.lib.zxing.config.ZXingLibConfig;
import jim.h.common.android.lib.zxing.integrator.IntentIntegrator;
import jim.h.common.android.lib.zxing.integrator.IntentResult;
import android.annotation.SuppressLint;
import android.app.ActionBar;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.content.res.Configuration;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.Bundle;
import android.os.Handler;
import android.os.StrictMode;
import android.support.v4.widget.DrawerLayout;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.FrameLayout;
import android.widget.LinearLayout;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import com.example.myfirstapp.amazonSigner.SignedRequestsHelper;
import com.example.myfirstapp.camerabase.CaptureActivity;
import com.ikimuhendis.ldrawer.ActionBarDrawerToggle;
```

```
import com.ikimuhendis.ldrawer.DrawerArrowDrawable;
import com.parse.ParseUser;

public class BaseActivity extends Activity {

    protected DrawerLayout mDrawerLayout;
    private ListView mDrawerList;
    private LinearLayout mLinear;

    FrameLayout frameLayout;

    private ActionBarDrawerToggle mDrawerToggle;
    private DrawerArrowDrawable drawerArrow;

    Context context = this;
    boolean gotCode = false;
    String thecode = "";
    String theTitle = "";
    String ocrText = "";
    Boolean scan = false;

    org.jsoup.nodes.Document doc1;

    private Handler        handler = new Handler();
    private ZXingLibConfig zxingLibConfig;

    @Override
    protected void onCreate(Bundle savedInstanceState) {

         // is network available
        if(!isNetworkAvailable()){
             showNetToast();
        }


        if (android.os.Build.VERSION.SDK_INT > 9) {
            StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy.Builder().permitAll().build();
            StrictMode.setThreadPolicy(policy);
        }

        // check for user
        ParseUser currentUser = ParseUser.getCurrentUser();
        if (currentUser == null) {
            loadLoginView();
        }

        // barcode scanner obj
        zxingLibConfig = new ZXingLibConfig();
        zxingLibConfig.useFrontLight = true;

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_sample);
        ActionBar ab = getActionBar();
        ab.setDisplayHomeAsUpEnabled(true);
        ab.setHomeButtonEnabled(true);

        // material drawer setup
        mDrawerLayout = (DrawerLayout) findViewById(R.id.drawer_layout);
        mDrawerList = (ListView) findViewById(R.id.navdrawer);
        mLinear = (LinearLayout) findViewById(R.id.header);
```

```
        drawerArrow = new DrawerArrowDrawable(this) {
            @Override
            public boolean isLayoutRtl() {
                return false;
            }
        };
        mDrawerToggle = new ActionBarDrawerToggle(this, mDrawerLayout,
            drawerArrow, R.string.drawer_open,
            R.string.drawer_close) {

            public void onDrawerClosed(View view) {
                super.onDrawerClosed(view);
                invalidateOptionsMenu();
            }

            public void onDrawerOpened(View drawerView) {
                super.onDrawerOpened(drawerView);
                invalidateOptionsMenu();
            }
        };
        mDrawerLayout.setDrawerListener(mDrawerToggle);
        mDrawerToggle.syncState();

        // drawer menu items
        String[] values = new String[]{
            "Inventory",
            "Scan Barcode",
            "Scan ISBN",
            "Categories",
            "Search",
            "Export",
            "Github"
        };
        // set drawer list
        ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
            android.R.layout.simple_list_item_1, android.R.id.text1,
values);
        mDrawerList.setAdapter(adapter);
        mDrawerList.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
            @SuppressLint("ResourceAsColor") @Override
            public void onItemClick(AdapterView<?> parent, View view,
                                    int position, long id) {
                switch (position) {
                    case 0:
                        Intent intent = new Intent(context,
ListItemActivity.class);
                        startActivity(intent);
                        break;
                    case 1:
                        IntentIntegrator.initiateScan(BaseActivity.this,
zxingLibConfig);
                        break;
                    case 2:
                        Intent intent2 = new Intent(context,
CaptureActivity.class);
                        intent2.putExtra("Uniqid","from_Main");
                        startActivityForResult(intent2, 11);
                        break;
                    case 3:
```

```
                        Intent intent3 = new Intent(context,
ListCategoryActivity.class);
                        startActivity(intent3);
                        break;
                    case 4:
                        Intent intent6 = new Intent(context,
SearchActivity.class);
                        startActivity(intent6);
                        break;
                    case 5:
                        Intent intent5 = new Intent(context,
ExportActivity.class);
                        startActivity(intent5);
                        break;
                    case 6:

                        break;
                }
            }
        });
    }

    //  get the xml from amazon
private String amazonReturnDetails(String barcode) throws
InvalidKeyException, NoSuchAlgorithmException,
ParserConfigurationException, SAXException, IOException{

        // create amazon request obj
        SignedRequestsHelper hq = new SignedRequestsHelper();

        Map<String, String> map = new HashMap<String, String>();

        map.put("AssociateTag", "PutYourAssociateTagHere");
        map.put("Keywords", barcode);
        map.put("Operation", "ItemSearch");
        map.put("SearchIndex", "All");
        map.put("ResponseGroup", "EditorialReview, Medium, Images");
        map.put("Service", "AWSECommerceService");
        map.put("Version", "2011-08-01");

        // pass map to helper obj to get the signed url
        String signedUrl = hq.sign(map);

        URL url = null;
        String inputLine;

        String theXML = "";

        // get the xml from the signed url and reutrn it
        try {
            url = new URL(signedUrl);
        } catch (MalformedURLException e) {
            e.printStackTrace();
        }
        BufferedReader in;
        try {
            URLConnection con = url.openConnection();
            con.setReadTimeout( 1000 ); //1 second
            in = new BufferedReader(new
InputStreamReader(con.getInputStream()));
            while ((inputLine = in.readLine()) != null) {
```

```
                theXML = inputLine;
            }
            in.close();

        } catch (IOException e) {
            e.printStackTrace();
        }

        return theXML;
    }

    // parse the xml to get the data
private static Map<String,String> parseXML(String xml) throws
ParserConfigurationException, SAXException, IOException{

    Map<String,String> xmlValues = new HashMap<String,String>();

    boolean isFirst = true;

    DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
    DocumentBuilder builder = factory.newDocumentBuilder();
    InputSource is = new InputSource(new StringReader(xml));
    Document doc = builder.parse(is);

    // normalize text representation
    doc.getDocumentElement ().normalize ();

    NodeList listOfPersons = doc.getElementsByTagName("ItemAttributes");

    for(int s=0; s<listOfPersons.getLength() ; s++){

        Node firstPersonNode = listOfPersons.item(s);
        if(firstPersonNode.getNodeType() == Node.ELEMENT_NODE && isFirst){

            Element firstPersonElement = (Element)firstPersonNode;

         // PRODUCT GROUP
            NodeList ProductGroupNodeElement =
firstPersonElement.getElementsByTagName("ProductGroup");
            if(ProductGroupNodeElement.getLength() == 1){
                Element ProductGroupElement =
(Element)ProductGroupNodeElement.item(0);

                NodeList ProductGroupText =
ProductGroupElement.getChildNodes();
                xmlValues.put("ProductGroup",
((Node)ProductGroupText.item(0)).getNodeValue().trim());
            }
            else
            {
                xmlValues.put("ProductGroup", "");
            }

            //TITLE
            NodeList lastNameList =
firstPersonElement.getElementsByTagName("Title");
            if(lastNameList.getLength() == 1){

            Element lastNameElement = (Element)lastNameList.item(0);

            NodeList textLNList = lastNameElement.getChildNodes();
```

```
                    xmlValues.put("Title",
((Node)textLNList.item(0)).getNodeValue().trim());
            }
            else
            {
                xmlValues.put("Title", "");
            }

            //MANUFACTURER
            NodeList firstNameList =
firstPersonElement.getElementsByTagName("Manufacturer");
            if(firstNameList.getLength() == 1){

            Element firstNameElement = (Element)firstNameList.item(0);

            NodeList textFNList = firstNameElement.getChildNodes();
                    xmlValues.put("Manufacturer",
((Node)textFNList.item(0)).getNodeValue().trim());
            }
            else
            {
                xmlValues.put("Manufacturer", "");
            }

            // AUTHOR
            NodeList authorNodeElement =
firstPersonElement.getElementsByTagName("Author");
            if(authorNodeElement.getLength() == 1){
                Element authorNameElement =
(Element)authorNodeElement.item(0);

                NodeList authorText = authorNameElement.getChildNodes();
                xmlValues.put("Author",
((Node)authorText.item(0)).getNodeValue().trim());
            }
            else
            {
                xmlValues.put("Author", "");
            }

            // ARTIST
            NodeList artistNodeElement =
firstPersonElement.getElementsByTagName("Artist");
            if(artistNodeElement.getLength() == 1){
                Element artistNameElement =
(Element)artistNodeElement.item(0);

                NodeList artistText = artistNameElement.getChildNodes();
                xmlValues.put("Artist",
((Node)artistText.item(0)).getNodeValue().trim());
            }
            else
            {
                xmlValues.put("Artist", "");
            }

            // RELEASE DATE
            NodeList releaseNodeElement =
firstPersonElement.getElementsByTagName("ReleaseDate");
            if(releaseNodeElement.getLength() == 1){
```

```
                Element releaseElement =
(Element)releaseNodeElement.item(0);

                NodeList releaseText = releaseElement.getChildNodes();
                xmlValues.put("Release",
((Node)releaseText.item(0)).getNodeValue().trim());
            }
            else
            {
                xmlValues.put("Release", "");
            }

            // PUBLICATION DATE
            NodeList publishNodeElement =
firstPersonElement.getElementsByTagName("PublicationDate");
            if(publishNodeElement.getLength() == 1){
                Element publishElement =
(Element)publishNodeElement.item(0);

                NodeList publishText = publishElement.getChildNodes();
                xmlValues.put("Publish",
((Node)publishText.item(0)).getNodeValue().trim());
            }
            else
            {
                xmlValues.put("Publish", "");
            }
            isFirst = false;

        }//end of if clause
    }//end of for loop with s var

    isFirst = true;

    NodeList listofDesc = doc.getElementsByTagName("EditorialReviews");

    for(int s=0; s<listofDesc.getLength() ; s++){

        Node firstPersonNode = listofDesc.item(s);
        if(firstPersonNode.getNodeType() == Node.ELEMENT_NODE && isFirst){

            Element firstPersonElement = (Element)firstPersonNode;

            //CONTENT
            NodeList firstNameList =
firstPersonElement.getElementsByTagName("Content");

                Element firstNameElement = (Element)firstNameList.item(0);

                NodeList textFNList = firstNameElement.getChildNodes();
                xmlValues.put("Content",
((Node)textFNList.item(0)).getNodeValue().trim());

            isFirst = false;
        }
    }

    isFirst = true;

    NodeList offerSummary = doc.getElementsByTagName("OfferSummary");
```

```
    for(int s=0; s<offerSummary.getLength() ; s++){

        Node firstPersonNode = offerSummary.item(s);
        if(firstPersonNode.getNodeType() == Node.ELEMENT_NODE && isFirst){

            Element firstPersonElement = (Element)firstPersonNode;

            //CONTENT
            NodeList firstNameList =
firstPersonElement.getElementsByTagName("FormattedPrice");

                Element firstNameElement = (Element)firstNameList.item(0);

                NodeList textFNList = firstNameElement.getChildNodes();
                xmlValues.put("Price",
((Node)textFNList.item(0)).getNodeValue().trim());

            isFirst = false;
        }
    }

    isFirst = true;

    NodeList mediumImage = doc.getElementsByTagName("LargeImage");

    for(int s=0; s<mediumImage.getLength() ; s++){

        Node medImageNode = mediumImage.item(s);
        if(medImageNode.getNodeType() == Node.ELEMENT_NODE && isFirst){

            Element medElement = (Element)medImageNode;

            //MEDIUM IMAGE
            NodeList firstNameList =
medElement.getElementsByTagName("URL");

                Element firstNameElement = (Element)firstNameList.item(0);

                NodeList textFNList = firstNameElement.getChildNodes();
                xmlValues.put("Image",
((Node)textFNList.item(0)).getNodeValue().trim());

            isFirst = false;
        }
    }
    // return the xml value map
    return xmlValues;
}


@Override
// get result from returned activity
protected void onActivityResult(int requestCode, int resultCode, Intent
data) {

    // first barcode scanner activity
    final Context context = this;
    super.onActivityResult(requestCode, resultCode, data);
    switch (requestCode) {
        case IntentIntegrator.REQUEST_CODE:
```

```
            IntentResult scanResult =
IntentIntegrator.parseActivityResult(requestCode,
                    resultCode, data);

            final String result = scanResult.getContents();

            if (result != null) {
                handler.post(new Runnable() {
                    @Override
                    public void run() {

                        String test = "";
                        String item = "";
                        String title = "";
                        String manufacturer = "";
                        String artist = "";
                        String author = "";
                        String description = "";
                        String price = "";
                        String image = "";
                        String publish = "";
                        String release = "";
                        try {
                            // get the xml data from the map
                            test = amazonReturnDetails(result);

                            item = parseXML(test).get("ProductGroup");
                            title = parseXML(test).get("Title");
                            manufacturer =
parseXML(test).get("Manufacturer");
                            artist =  parseXML(test).get("Artist");
                            author =  parseXML(test).get("Author");
                            description =  parseXML(test).get("Content");
                            price =  parseXML(test).get("Price");
                            image =  parseXML(test).get("Image");
                            publish =  parseXML(test).get("Publish");
                            release =  parseXML(test).get("Release");

                            doc1 = Jsoup.parse(description);
                            description = doc1.text();
                        } catch (Exception e) {
                            // TODO Auto-generated catch block
                            e.printStackTrace();
                            System.out.println("ERROR");
                        }

                        Intent intent = new Intent(context,
AddItemActivity.class);

                        intent.putExtra("Uniqid","from_Main");
                        intent.putExtra("title", title);
                        intent.putExtra("description", description);
                        intent.putExtra("category", item);
                        intent.putExtra("price", price);
                        intent.putExtra("image", image);
                        intent.putExtra("manufacturer", manufacturer);

                        // check what type of item, add more fields if
necessary
                        if(item != null){
```

```
                            if(item.equals("Book") ||
item.equals("eBooks")){

                                intent.putExtra("author", author);
                                intent.putExtra("release", publish);
                            }
                            if(item.equals("Music")){
                                intent.putExtra("author", artist);
                                intent.putExtra("release", release);
                            }
                            if(item.equals("DVD") ||
item.equals("VideoGames")){
                                intent.putExtra("release", release);
                            }
                        }
                        else{
                            intent.putExtra("notFound", "No item found.
Please enter it manually");
                        }
                        startActivity(intent);
                    }
                });
            }
            break;
            // OCR scanner
        case 11:
            if(data != null){
            // replace letters in the text
            ocrText = data.getStringExtra("ocr");
            ocrText = ocrText.replaceAll("[\\s\\-()]", "");

            if (ocrText != null) {
                handler.post(new Runnable() {
                    @Override
                    public void run() {

                        String test = "";

                        String item = "";
                        String title = "";
                        String manufacturer = "";
                        String artist = "";
                        String author = "";
                        String description = "";
                        String price = "";
                        String image = "";
                        String publish = "";
                        String release = "";
                        try {
                            // get the data from the xml
                            ocrText = ocrText.replaceAll("[\\s\\-()]", "");

                            test = amazonReturnDetails(ocrText);
                            System.out.println(test);
                            item = parseXML(test).get("ProductGroup");
                            title = parseXML(test).get("Title");
                            manufacturer =
parseXML(test).get("Manufacturer");
                            artist =  parseXML(test).get("Artist");
                            author =  parseXML(test).get("Author");
                            description =  parseXML(test).get("Content");
```

```
                            price =  parseXML(test).get("Price");
                            image =  parseXML(test).get("Image");
                            publish =  parseXML(test).get("Publish");
                            release =  parseXML(test).get("Release");

                            doc1 = Jsoup.parse(description);
                            description = doc1.text();
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                        // add the data to the intent
                        Intent intent = new Intent(context,
AddItemActivity.class);

                        intent.putExtra("Uniqid","from_Main");
                        intent.putExtra("title", title);
                        intent.putExtra("description", description);
                        intent.putExtra("category", item);
                        intent.putExtra("price", price);
                        intent.putExtra("image", image);
                        intent.putExtra("manufacturer", manufacturer);
                        intent.putExtra("publish", publish);

                        // check for item type and add other details if
needed
                        if(item != null){
                            if(item.equals("Book") ||
item.equals("eBooks")){
                                intent.putExtra("author", author);
                                intent.putExtra("release", publish);
                            }
                            if(item.equals("Music")){
                                intent.putExtra("artist", artist);
                                intent.putExtra("release", release);
                            }
                            if(item.equals("DVD") ||
item.equals("VideoGames")){
                                intent.putExtra("release", release);
                            }
                        }
                        else
                        {
                            intent.putExtra("notFound", "No item found for
isbn: " + ocrText + ". Please check isbn number or enter item manually");
                        }
                        // start the activity
                        startActivity(intent);
                    }
                });
            }
            }
            break;
        default:
    }
}

// load the login view if logged out
private void loadLoginView() {
    Intent intent = new Intent(this, LoginActivity.class);
    intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
    intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK);
```

```
    startActivity(intent);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is
present.
    getMenuInflater().inflate(R.menu.main_menu, menu);
    return true;
}

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        if (item.getItemId() == android.R.id.home) {
            if (mDrawerLayout.isDrawerOpen(mLinear)) {
                mDrawerLayout.closeDrawer(mLinear);
            } else {
                mDrawerLayout.openDrawer(mLinear);
            }
        }
        if (item.getItemId() == R.id.action_logout) {
            ParseUser.logOut();
            loadLoginView();
        }
        return super.onOptionsItemSelected(item);
    }

    @Override
    protected void onPostCreate(Bundle savedInstanceState) {
        super.onPostCreate(savedInstanceState);
        mDrawerToggle.syncState();
    }

    @Override
    public void onConfigurationChanged(Configuration newConfig) {
        super.onConfigurationChanged(newConfig);
        mDrawerToggle.onConfigurationChanged(newConfig);
    }

    // check if network available
    private boolean isNetworkAvailable() {
        ConnectivityManager connectivityManager
              = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo activeNetworkInfo =
connectivityManager.getActiveNetworkInfo();
        return activeNetworkInfo != null &&
activeNetworkInfo.isConnected();
    }

    // show network error message
    private void showNetToast(){
         Toast toast = Toast.makeText(context, "No Internet Connection",
Toast.LENGTH_LONG);
         View view = toast.getView();
         view.setBackgroundResource(R.color.toast_nointernet_color);
         TextView text = (TextView)
view.findViewById(android.R.id.message);
         /*here you can do anything with text*/
         toast.show();
}}
```

Export.java

```
package com.example.myfirstapp;

import java.io.File;
import java.io.FileWriter;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

import com.opencsv.CSVWriter;
import com.parse.FindCallback;
import com.parse.ParseException;
import com.parse.ParseObject;
import com.parse.ParseQuery;
import com.parse.ParseUser;

import android.app.Activity;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.os.Environment;
import android.text.TextUtils;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.ArrayAdapter;

public class ExportActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        deleteItem();

        // create a folder on the device
        File exportDir = new File(Environment.getExternalStorageDirectory()
                .getPath(), "/sqliteDB");

        // create a new file
        File file = new File(exportDir, "exportInv.csv");

        // pass the file to the intent
        Uri u1 = Uri.fromFile(file);
        Intent sendIntent = new Intent(Intent.ACTION_SEND);
        sendIntent.putExtra(Intent.EXTRA_SUBJECT, "Inventory CSV");
        sendIntent.putExtra(Intent.EXTRA_STREAM, u1);
        sendIntent.setType("text/richtext");
        startActivity(sendIntent);
        finish();
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.export, menu);
        return true;
```

```
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);
    }

    public void deleteItem(){
        final List<String> title_list = new ArrayList<String>();
        final List<String> tester = new ArrayList<String>();
        final List<String[]> database = new ArrayList<String[]>();

        // query
        ParseQuery<ParseObject> query = ParseQuery.getQuery("Inventory");
        query.whereEqualTo("author", ParseUser.getCurrentUser());
        query.setLimit(1000);
        query.findInBackground(new FindCallback<ParseObject>() {

            @SuppressWarnings("unchecked")
            @Override
            public void done(List<ParseObject> postList, ParseException e)
{
                setProgressBarIndeterminateVisibility(false);
                if (e == null) {
                    for(ParseObject message : postList)
                    {
                         database.add(new String[]
{message.get("title").toString(), message.get("description").toString(),
message.get("category").toString()});
                    }
                    // header for csv file
                    String[] header= new
String[]{"Title","Description","Category"};

                    // export dir
                    File exportDir = new
File(Environment.getExternalStorageDirectory()
                            .getPath(), "/sqliteDB");

                    // create a folder in none exists
                    if (!exportDir.exists())
                    {
                        exportDir.mkdirs();
                    }

                    // create file
                    File file = new File(exportDir, "exportInv.csv");

                    try{
                        file.createNewFile();
                        CSVWriter csvWrite = new CSVWriter(new
FileWriter(file));
                        csvWrite.writeNext(header);
                        csvWrite.writeAll(database);
```

```
                    csvWrite.close();
                }
                catch(Exception ex){
                    // error
                }
            } else {
                Log.d(getClass().getSimpleName(), "Error: " +
e.getMessage());
                System.out.println(e.getMessage());
            }
        }
    });
  }
}
```

FoundItemsActivity.java

```
package com.example.myfirstapp;

import java.util.ArrayList;
import java.util.List;
import com.parse.FindCallback;
import com.parse.ParseObject;
import com.parse.ParseQuery;
import com.parse.ParseUser;
import com.parse.ParseException;
import android.app.ListActivity;
import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

public class FoundItemActivity extends ListActivity {

    private List<ItemDetails> items;
    private ArrayList<String> titleList;
    private String title;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.list_item);

        // check if network available
        if(!isNetworkAvailable()){

            showNetToast();
        }
        // array list of item details
        items = new ArrayList<ItemDetails>();
        //Get the bundle
        Bundle bundle = getIntent().getExtras();

        //Extract the data
        if(bundle != null){
            title = bundle.getString("term");
            title = title.toLowerCase();
        }

        // search the item
        searchItem(title);

        titleList = new ArrayList<String>();

        // array for list view
```

```
        ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
R.layout.list_item_layout, titleList);
        setListAdapter(adapter);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.found_item, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);
    }



    // search for the item based on the title
    private void searchItem(String theTitle) {

        // connect to database
        ParseQuery<ParseObject> query = ParseQuery.getQuery("Inventory");
        query.whereEqualTo("author", ParseUser.getCurrentUser());
        query.whereContains("title", title);

        setProgressBarIndeterminateVisibility(true);

        query.findInBackground(new FindCallback<ParseObject>() {

            @SuppressWarnings("unchecked")
            @Override
            public void done(List<ParseObject> itemList, ParseException e)
{
                setProgressBarIndeterminateVisibility(false);
                if (e == null) {
                    // If there are results, update the list of items
                    // and notify the adapter
                    for (ParseObject ob : itemList) {
                        titleList.add(ob.getString("title"));
                        ItemDetails note = new
ItemDetails(ob.getObjectId(), ob.getString("title"),
ob.getString("description"), ob.getString("category"),
ob.getParseFile("photo"), ob.getString("releaseDate"),
ob.getString("author_artist"));
                        items.add(note);
                    }
                    ((ArrayAdapter<ItemDetails>)
getListAdapter()).notifyDataSetChanged();
                } else {
                    Log.d(getClass().getSimpleName(), "Error: " +
e.getMessage());
```

```
                }
            }
        });
    }

    // on item click
    @Override
    protected void onListItemClick(ListView l, View v, int position, long
id) {

        ItemDetails theItems = items.get(position);

        Intent intent = new Intent(this, AddItemActivity.class);
        intent.putExtra("Uniqid","from_List_Activity");
        intent.putExtra("itemId", theItems.getId());
        intent.putExtra("itemTitle", theItems.getTitle());
        intent.putExtra("itemContent", theItems.getContent());
        intent.putExtra("itemCategory", theItems.getCategory());
        intent.putExtra("itemImage", theItems.getPhoto().getUrl());
        startActivity(intent);
        finish();
    }

    // check for network
    private boolean isNetworkAvailable() {
        ConnectivityManager connectivityManager
              = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo activeNetworkInfo =
connectivityManager.getActiveNetworkInfo();
        return activeNetworkInfo != null &&
activeNetworkInfo.isConnected();
    }

    private void showNetToast(){
         Toast toast = Toast.makeText(this, "No Internet Connection",
Toast.LENGTH_LONG);
         View view = toast.getView();
         view.setBackgroundResource(R.color.toast_nointernet_color);
         TextView text = (TextView)
view.findViewById(android.R.id.message);
         /*here you can do anything with text*/
         toast.show();

    }
}
```

InventoryApplication.java

```
package com.example.myfirstapp;

import com.parse.Parse;
import android.app.Application;

public class InventoryApplication extends Application {

    @Override
    public void onCreate() {
        super.onCreate();

        // allow connection to cloud
        Parse.initialize(this, "saIQWlbzHC5G90hwMY0FdjnuN2dQInrRGMwuCMjy",
"bN0TkD4r0ELzZaWX9zlNqoYJnnWUDyK2qSsOTDQT");
    }
}
```

ItemDetails.java

```
package com.example.myfirstapp;

import com.parse.ParseFile;

/*
 * Create a class that will hold our Item data.
 * Contains item id, details, title, category fields and their getters and
setters.
 */

public class ItemDetails {

    private String id;
    private String title;
    private String details;
    private String category;
    private String author_artist;
    private String releaseDate;
    private String image;
    private ParseFile file;

    // constructors
    ItemDetails(String theId, String theTitle, String theContent, String
theCategory, ParseFile theImage, String theAuthor_artist, String
theReleaseDate) {

        id = theId;
        title = theTitle;
        details = theContent;
        category = theCategory;
        author_artist = theAuthor_artist;
        releaseDate = theReleaseDate;
        file = theImage;
    }
    ItemDetails(String theId, String theTitle, String theContent, String
theCategory, String theAuthor_artist, String theReleaseDate) {

        id = theId;
        title = theTitle;
        details = theContent;
        category = theCategory;
        author_artist = theAuthor_artist;
        releaseDate = theReleaseDate;
    }

        public String getId() {
            return id;
        }
        public void setId(String id) {
            this.id = id;
        }
        public String getTitle() {
            return title;
        }
        public void setTitle(String title) {
            this.title = title;
        }
```

```
    public String getContent() {
        return details;
    }
    public void setContent(String content) {
        this.details = content;
    }
    public String getCategory() {
        return category;
    }
    public void setCategory(String category) {
        this.category = category;
    }
    public String getImage() {
        return image;
    }
    public void setImage(String image) {
        this.image = image;
    }
    public ParseFile getPhoto() {
        return file;
    }
    public void setPhoto(ParseFile file) {
        this.file = file;
    }
    public String getAuthor_Artist() {
        return author_artist;
    }
    public void setAuthor_Artist(String theAuthor_artist) {
        this.author_artist = theAuthor_artist;
    }
    public String getReleaseDate() {
        return releaseDate;
    }
    public void setReleaseDate(String theReleaseDate) {
        this.releaseDate = theReleaseDate;
    }
    @Override
    public String toString() {
        return this.getTitle();
    }
}
```

ListCategoryActivity.java

```
package com.example.myfirstapp;

import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;
import com.parse.FindCallback;
import com.parse.ParseException;
import com.parse.ParseObject;
import com.parse.ParseQuery;
import com.parse.ParseUser;
import android.app.ListActivity;
import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

public class ListCategoryActivity extends ListActivity {

    private ArrayList<String> categoryList;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.list_item);

        if(!isNetworkAvailable()){
            showNetToast();
        }

        categoryList = new ArrayList<String>();
        ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
R.layout.list_item_layout, categoryList);
        setListAdapter(adapter);
        refreshItemList();
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.list_category, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
```

```
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);
    }

    private void refreshItemList() {

        ParseQuery<ParseObject> query = ParseQuery.getQuery("Inventory");
        query.whereEqualTo("author", ParseUser.getCurrentUser());
        query.findInBackground(new FindCallback<ParseObject>() {
            public void done(List<ParseObject> itemList, ParseException e)
{
                if (e == null) {
                    for (ParseObject ob : itemList) {
                        categoryList.add(ob.getString("category"));
                    }
                    Set<String> set = new HashSet<String>();
                    set.addAll(categoryList);
                    categoryList.clear();
                    categoryList.addAll(set);
                    ((ArrayAdapter<ItemDetails>)
getListAdapter()).notifyDataSetChanged();
                } else {
                    Log.d("message", "Error: " + e.getMessage());
                }
            }
        });
    }
       protected void onListItemClick(ListView l, View v, int position,
long id) {
   super.onListItemClick(l, v, position, id);
   Object o = this.getListAdapter().getItem(position);
   String pen = o.toString();

   Intent intent = new Intent(this, ListCategoryItemActivity.class);
   intent.putExtra("theCategory",pen);
   startActivity(intent);
   }
    private boolean isNetworkAvailable() {
        ConnectivityManager connectivityManager
              = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo activeNetworkInfo =
connectivityManager.getActiveNetworkInfo();
        return activeNetworkInfo != null &&
activeNetworkInfo.isConnected();
    }
      private void showNetToast(){
         Toast toast = Toast.makeText(this, "No Internet Connection",
Toast.LENGTH_LONG);
         View view = toast.getView();
         view.setBackgroundResource(R.color.toast_nointernet_color);
         TextView text = (TextView)
view.findViewById(android.R.id.message);
         /*here you can do anything with text*/
         toast.show();
    }
}
```

ListCategoryItemActivity.java

```
package com.example.myfirstapp;

import java.util.ArrayList;
import java.util.List;
import com.parse.FindCallback;
import com.parse.ParseObject;
import com.parse.ParseQuery;
import com.parse.ParseUser;
import android.app.ListActivity;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.Window;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import com.parse.ParseException;

public class ListCategoryItemActivity extends ListActivity {

    // declare a class variable that will hold a list of items.
    private List<ItemDetails> items;
    String theCategory = "";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_INDETERMINATE_PROGRESS);
        setContentView(R.layout.list_item);

        Intent intent = this.getIntent();
        theCategory = intent.getStringExtra("theCategory");

        items = new ArrayList<ItemDetails>();
        // adapter for holding items
        ArrayAdapter<ItemDetails> adapter = new
ArrayAdapter<ItemDetails>(this, R.layout.list_item_layout, items);
        setListAdapter(adapter);

        refreshItemList();
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.list_category_item, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
```

```
        }
        return super.onOptionsItemSelected(item);
    }

    // show the list of items
    private void refreshItemList() {
        // database query
        ParseQuery<ParseObject> query = ParseQuery.getQuery("Inventory");
        query.whereEqualTo("author", ParseUser.getCurrentUser());
        query.whereEqualTo("category", theCategory);
        setProgressBarIndeterminateVisibility(true);

        query.findInBackground(new FindCallback<ParseObject>() {

            @SuppressWarnings("unchecked")
            @Override
            public void done(List<ParseObject> postList, ParseException e)
{
                setProgressBarIndeterminateVisibility(false);
                if (e == null) {
                    // If there are results, update the list of items
                    // and notify the adapter
                    items.clear();
                    for (ParseObject post : postList) {
                        ItemDetails note = new
ItemDetails(post.getObjectId(), post.getString("title"),
post.getString("description"), post.getString("category"),
post.getParseFile("photo"), post.getString("author_artist"),
post.getString("releaseDate"));
                        items.add(note);
                    }
                    ((ArrayAdapter<ItemDetails>)
getListAdapter()).notifyDataSetChanged();
                } else {
                    Log.d(getClass().getSimpleName(), "Error: " +
e.getMessage());
                }
            }
        });
    }
       @Override
    protected void onListItemClick(ListView l, View v, int position, long
id) {

        ItemDetails theItems = items.get(position);

        Intent intent = new Intent(this, AddItemActivity.class);
        intent.putExtra("Uniqid","from_List_Activity");
        intent.putExtra("itemId", theItems.getId());
        intent.putExtra("itemTitle", theItems.getTitle());
        intent.putExtra("itemContent", theItems.getContent());
        intent.putExtra("itemCategory", theItems.getCategory());
        intent.putExtra("itemImage", theItems.getPhoto().getUrl());
        startActivity(intent);
        finish();
    }
}
```

ListItemActivity.java

```
package com.example.myfirstapp;

import java.util.ArrayList;
import java.util.List;
import com.parse.FindCallback;
import com.parse.ParseObject;
import com.parse.ParseQuery;
import com.parse.ParseUser;
import android.app.ListActivity;
import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.Bundle;
import android.util.Log;
import android.view.ContextMenu;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.view.Window;
import android.view.ContextMenu.ContextMenuInfo;
import android.widget.AdapterView;
import android.widget.AdapterView.AdapterContextMenuInfo;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import com.parse.ParseException;

public class ListItemActivity extends ListActivity {

    // declare a class variable that will hold a list of items.
    private List<ItemDetails> items;
    boolean deleted = false;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_INDETERMINATE_PROGRESS);
        setContentView(R.layout.list_item);

        // if network not available show message
        if(!isNetworkAvailable()){
            showNetToast();
        }

        registerForContextMenu(getListView());

        /* create an adapter which manages the data model and adapts it to
the individual rows in the list view.
         * Use the given adapter and override the objects toString() method
so that it gives the title of the item.
         */

        items = new ArrayList<ItemDetails>();
```

```
        ArrayAdapter<ItemDetails> adapter = new
ArrayAdapter<ItemDetails>(this, R.layout.list_item_layout, items);
        setListAdapter(adapter);

        refreshItemList();
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }

    // action bar option click
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();

        switch (id) {
           case R.id.action_refresh: {
            refreshItemList();
            break;
        }
           // create new item
        case R.id.action_new: {
            Intent intent = new Intent(this, AddItemActivity.class);
            intent.putExtra("Uniqid","from_List_Activity_Menu");
            startActivity(intent);
            break;
        }
        case R.id.action_settings: {
            // Do something when user selects Settings from Action Bar
overlay
            break;
        }
        }
        return super.onOptionsItemSelected(item);
    }

    // long press to find which item
    public void onCreateContextMenu(ContextMenu menu, View v,
ContextMenuInfo menuInfo) {
        super.onCreateContextMenu(menu, v, menuInfo);
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.context_menu, menu);
        long
pos=getListAdapter().getItemId(((AdapterView.AdapterContextMenuInfo)menuInf
o).position);
        ItemDetails theItems = items.get((int) pos);
        theItems = items.get((int) pos);
    }
    // long press item options - delete/refresh
    public boolean onContextItemSelected(MenuItem item) {
        AdapterContextMenuInfo info = (AdapterContextMenuInfo)
item.getMenuInfo();
        ItemDetails theItems;
```

```
        switch(item.getItemId()){
        case R.id.delete:
            theItems = items.get(info.position);
            String itemId = theItems.getId();
            String itemName = theItems.getTitle();
            deleteItem(itemId, itemName);
            ParseObject.createWithoutData("Inventory",
itemId).deleteEventually();
            Toast.makeText(ListItemActivity.this, "You have deleted item: "
+ " " + itemName , Toast.LENGTH_LONG).show();
            deleted = true;
            if (deleted = true){
                refreshItemList();
                deleted = false;
            }
            break;
        case R.id.edit:
            theItems = items.get(info.position);
            System.out.println("Edit item: " + info.position);
            break;
        default:
            refreshItemList();
            break;
        }
              return super.onContextItemSelected(item);
        }

        // refresh item list / show items
        private void refreshItemList() {

        ParseQuery<ParseObject> query = ParseQuery.getQuery("Inventory");
        query.whereEqualTo("author", ParseUser.getCurrentUser());

        setProgressBarIndeterminateVisibility(true);

        query.findInBackground(new FindCallback<ParseObject>() {

            @SuppressWarnings("unchecked")
            @Override
            public void done(List<ParseObject> itemList, ParseException e)
{
                setProgressBarIndeterminateVisibility(false);
                if (e == null) {
                    // If there are results, update the list of items
                    // and notify the adapter
                    items.clear();
                    for (ParseObject theItem : itemList) {
                        ItemDetails myitems = new
ItemDetails(theItem.getObjectId(), theItem.getString("title"),
theItem.getString("description"), theItem.getString("category"),
theItem.getParseFile("photo"), theItem.getString("author_artist"),
theItem.getString("releaseDate"));
                        items.add(myitems);
                    }
                    ((ArrayAdapter<ItemDetails>)
getListAdapter()).notifyDataSetChanged();
                } else {
                    Log.d(getClass().getSimpleName(), "Error: " +
e.getMessage());
                }
            }
```

```
        });
    }

    protected void onListItemClick(ListView l, View v, int position, long
id) {

        ItemDetails theItems = items.get(position);

        Intent intent = new Intent(this, AddItemActivity.class);
        intent.putExtra("Uniqid","from_List_Activity");
        intent.putExtra("itemId", theItems.getId());
        intent.putExtra("itemTitle", theItems.getTitle());
        intent.putExtra("itemContent", theItems.getContent());
        intent.putExtra("itemCategory", theItems.getCategory());
        intent.putExtra("itemImage", theItems.getPhoto().getUrl());
        intent.putExtra("itemAuthor_artist", theItems.getAuthor_Artist());
        intent.putExtra("itemReleaseDate", theItems.getReleaseDate());

        startActivity(intent);
    }
    public void deleteItem(String itemId, final String itemName){

        ParseQuery<ParseObject> query = ParseQuery.getQuery("Inventory");
        query.whereEqualTo("objectid", itemId);
        query.findInBackground(new FindCallback<ParseObject>() {

            @SuppressWarnings("unchecked")
            @Override
            public void done(List<ParseObject> postList, ParseException e)
{
                setProgressBarIndeterminateVisibility(false);
                if (e == null) {
                    // If there are results, update the list of posts
                    // and notify the adapter
                    // iterate over all messages and delete them
                    for(ParseObject message : postList)
                    {
                         message.deleteEventually();
                    }
                } else {
                    Log.d(getClass().getSimpleName(), "Error: " +
e.getMessage());
                }
            }
        });
    }
    private boolean isNetworkAvailable() {
        ConnectivityManager connectivityManager
              = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo activeNetworkInfo =
connectivityManager.getActiveNetworkInfo();
        return activeNetworkInfo != null &&
activeNetworkInfo.isConnected();
    }
    private void showNetToast(){
         Toast toast = Toast.makeText(this, "No Internet Connection",
Toast.LENGTH_LONG);
         View view = toast.getView();
         view.setBackgroundResource(R.color.toast_nointernet_color);
```

```
        TextView text = (TextView)
view.findViewById(android.R.id.message);
        toast.show();
    }}
```

LoginActivity.java

```
package com.example.myfirstapp;

import com.parse.LogInCallback;
import com.parse.ParseUser;
import com.parse.ParseException;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.Window;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class LoginActivity extends Activity {

    protected EditText usernameEditText;
    protected EditText passwordEditText;
    protected Button loginButton;

    protected TextView signUpTextView;
    protected TextView resetPasswordTextView;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_INDETERMINATE_PROGRESS);
        setContentView(R.layout.activity_login);

        // set up textviews
        signUpTextView = (TextView)findViewById(R.id.signUpText);
        resetPasswordTextView =
(TextView)findViewById(R.id.forgottenPasswordText);
        usernameEditText = (EditText)findViewById(R.id.usernameField);

        // text boxes
        passwordEditText = (EditText)findViewById(R.id.passwordField);
        loginButton = (Button)findViewById(R.id.loginButton);

        // sign up text view click
        signUpTextView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(LoginActivity.this,
SignUpActivity.class);
                startActivity(intent);
            }
        });

        // reset password text view click
```

```
        resetPasswordTextView.setOnClickListener(new View.OnClickListener()
{
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(LoginActivity.this,
PasswordReset.class);
                startActivity(intent);
            }
        });

         // login button
        loginButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

usernameEditText.setInputType(android.text.InputType.TYPE_CLASS_TEXT |
android.text.InputType.TYPE_TEXT_FLAG_MULTI_LINE);
                String username = usernameEditText.getText().toString();
                String password = passwordEditText.getText().toString();

                username = username.trim();
                password = password.trim();

                // validation
                if (username.isEmpty() || password.isEmpty()) {
                    AlertDialog.Builder builder = new
AlertDialog.Builder(LoginActivity.this);
                    builder.setMessage(R.string.login_error_message)
                        .setTitle(R.string.login_error_title)
                        .setPositiveButton(android.R.string.ok, null);
                    AlertDialog dialog = builder.create();
                    dialog.show();
                }
                else {
                    setProgressBarIndeterminateVisibility(true);
                    // login
                    ParseUser.logInInBackground(username, password, new
LogInCallback() {
                        @Override
                        public void done(ParseUser user, ParseException e)
{
                            setProgressBarIndeterminateVisibility(false);

                            if (e == null) {
                                // Success!
                                Intent intent = new
Intent(LoginActivity.this, BaseActivity.class);

intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);

intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK);
                                startActivity(intent);
                            }
                            else {
                                // Fail
                                AlertDialog.Builder builder = new
AlertDialog.Builder(LoginActivity.this);
                                builder.setMessage(e.getMessage())
                                    .setTitle(R.string.login_error_title)
                                    .setPositiveButton(android.R.string.ok,
null);
```

```
                                AlertDialog dialog = builder.create();
                                dialog.show();
                            }
                        }
                    });
                }
            }
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.login, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);
    }
}
```

PasswordReset.java

```
package com.example.myfirstapp;

import com.parse.ParseException;
import com.parse.ParseUser;
import com.parse.RequestPasswordResetCallback;
import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class PasswordReset extends Activity {
    // variables
    private EditText emailEditBox;
    private Button resetButton;
    private String email;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_password_reset);

        // set up boxes
        emailEditBox = (EditText) findViewById(R.id.emailField);
        resetButton = (Button)findViewById(R.id.resetPasswordButton);

        // reset button click
        resetButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                email = emailEditBox.getText().toString();
                // send the email
                ParseUser.requestPasswordResetInBackground(email, new
RequestPasswordResetCallback() {
                    public void done(ParseException e) {
                        if (e == null) {
                            // An email was successfully sent with reset
instructions.
                            Toast.makeText(getApplicationContext(),
"Password reset instructions sent", Toast.LENGTH_SHORT).show();
                        } else {
                            // Something went wrong. Look at the
ParseException to see what's up.
                            Toast.makeText(getApplicationContext(), "Error
sending Password reset instructions", Toast.LENGTH_SHORT).show();
                        }
                    }
                });
            }
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
```

```
        getMenuInflater().inflate(R.menu.password_reset, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);
    }
}
```

SearchActivity.java

```
package com.example.myfirstapp;

import java.util.ArrayList;
import java.util.List;
import com.parse.FindCallback;
import com.parse.ParseException;
import com.parse.ParseObject;
import com.parse.ParseQuery;
import com.parse.ParseUser;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.EditText;
import android.widget.LinearLayout;

public class SearchActivity extends BaseActivity {

    // array list for storing item details
    ArrayList<ItemDetails> itemList = new ArrayList<ItemDetails>();
    private List<ItemDetails> items;
    Context context = this;

    private EditText seachEditText;
    private String itemTitle;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // drawer setup
        LayoutInflater inflater = (LayoutInflater) this
                .getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        View contentView = inflater.inflate(R.layout.activity_search, null,
false);
        mDrawerLayout.addView(contentView, 0);

        // clear the background
        LinearLayout  linearLayout = (LinearLayout)
findViewById(R.id.linearLayoutid);
         linearLayout.setBackgroundResource(0);

        seachEditText = (EditText) findViewById(R.id.myFilter);

        // search button
        View btnSearch = findViewById(R.id.search_button);
        btnSearch.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                itemTitle = seachEditText.getText().toString();
                Intent intent = new Intent(context,
FoundItemActivity.class);
                intent.putExtra("term", itemTitle);
                startActivity(intent);
            }
```

```
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.search, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);
    }

    // search item via title
    private void searchItem(String theTitle) {

        // query
        ParseQuery<ParseObject> query = ParseQuery.getQuery("Inventory");
        query.whereEqualTo("author", ParseUser.getCurrentUser());
        query.whereEqualTo("title", theTitle);

        // progress spinner
        setProgressBarIndeterminateVisibility(true);

        query.findInBackground(new FindCallback<ParseObject>() {

            @SuppressWarnings("unchecked")
            @Override
            public void done(List<ParseObject> postList, ParseException e)
{
                setProgressBarIndeterminateVisibility(false);
                if (e == null) {
                    // If there are results, update the list of items
                    // and notify the adapter
                    items.clear();
                    for (ParseObject post : postList) {
                        ItemDetails note = new
ItemDetails(post.getObjectId(), post.getString("title"),
post.getString("description"), post.getString("category"),
post.getParseFile("photo"),
post.getString("author_artist"),post.getString("releaseDate"));
                        items.add(note);
                    }
                } else {
                    Log.d(getClass().getSimpleName(), "Error: " +
e.getMessage());
                }
            }
        });
    }}
```

SignupActivity.java

```
package com.example.myfirstapp;

import com.parse.ParseUser;
import com.parse.SignUpCallback;
import com.parse.ParseException;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.Window;
import android.widget.Button;
import android.widget.EditText;

public class SignUpActivity extends Activity {
    protected EditText usernameEditText;
    protected EditText passwordEditText;
    protected EditText emailEditText;
    protected Button signUpButton;
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        requestWindowFeature(Window.FEATURE_INDETERMINATE_PROGRESS);
        getActionBar().setDisplayHomeAsUpEnabled(true);

        setContentView(R.layout.activity_sign_up);

        usernameEditText = (EditText)findViewById(R.id.usernameField);
        passwordEditText = (EditText)findViewById(R.id.passwordField);
        emailEditText = (EditText)findViewById(R.id.emailField);

        signUpButton = (Button)findViewById(R.id.signupButton);

        // signup button
        signUpButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // set keyboard lowercase

usernameEditText.setInputType(android.text.InputType.TYPE_CLASS_TEXT |
android.text.InputType.TYPE_TEXT_FLAG_MULTI_LINE);
                String username = usernameEditText.getText().toString();
                String password = passwordEditText.getText().toString();
                String email = emailEditText.getText().toString();

                username = username.trim();
                password = password.trim();
                email = email.trim();
                if (username.isEmpty() || password.isEmpty() ||
email.isEmpty()) {
                    AlertDialog.Builder builder = new
AlertDialog.Builder(SignUpActivity.this);
                    builder.setMessage(R.string.signup_error_message)
                        .setTitle(R.string.signup_error_title)
                        .setPositiveButton(android.R.string.ok, null);
                    AlertDialog dialog = builder.create();
                    dialog.show();
                }
```

```
                else {
                    // progress spinner on
                    setProgressBarIndeterminateVisibility(true);

                    // create new user with the supplied details
                    ParseUser newUser = new ParseUser();
                    newUser.setUsername(username);
                    newUser.setPassword(password);
                    newUser.setEmail(email);
                    newUser.signUpInBackground(new SignUpCallback() {
                        @Override
                        public void done(ParseException e) {
                                // progress spinner off
                            setProgressBarIndeterminateVisibility(false);
                            if (e == null) {
                                // Success!
                                Intent intent = new
Intent(SignUpActivity.this, BaseActivity.class);

intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);

intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK);
                                startActivity(intent);
                            }
                            else {
                                // fail msg
                                AlertDialog.Builder builder = new
AlertDialog.Builder(SignUpActivity.this);
                                builder.setMessage(e.getMessage())
                                    .setTitle(R.string.signup_error_title)
                                    .setPositiveButton(android.R.string.ok,
null);

                                AlertDialog dialog = builder.create();
                                dialog.show();
                            }
                        }
                    });
                }
            }
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.sign_up, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);}}
```

Website

Index.php

```
<?php
include('processlogin.php'); // Includes Login Script

if(isset($_SESSION['login_user'])){
$log = True;
}
else{
$log = False;
}
?>

<!-- index page -->
<!doctype html>
<html lang="en">
<head>
    <meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta name="description" content="A layout example that shows off a
responsive product landing page.">
    <title>Landing Page &ndash; Layout Examples &ndash; Pure</title>

<link rel="stylesheet" href="http://yui.yahooapis.com/pure/0.6.0/pure-
min.css">
<!--[if lte IE 8]>
      <link rel="stylesheet"
href="http://yui.yahooapis.com/pure/0.6.0/grids-responsive-old-ie-min.css">
  <![endif]-->
<!--[if gt IE 8]><!-->
      <link rel="stylesheet"
href="http://yui.yahooapis.com/pure/0.6.0/grids-responsive-min.css">
  <!--<![endif]-->

<link rel="stylesheet" href="http://netdna.bootstrapcdn.com/font-
awesome/4.0.3/css/font-awesome.css">

    <!--[if lte IE 8]>
        <link rel="stylesheet" href="css/layouts/marketing-old-ie.css">
    <![endif]-->
    <!--[if gt IE 8]><!-->
        <link rel="stylesheet" href="css/layouts/marketing.css">
    <!--<![endif]-->

</head>
<body>

<div class="header">
    <div class="home-menu pure-menu pure-menu-horizontal pure-menu-fixed">
        <a class="pure-menu-heading" href="">My Inventory</a>
        <!-- display home page depending on logged in status -->
        <?php
        if(!$log)
        {
        ?>
        <ul class="pure-menu-list">
            <li class="pure-menu-item"><a href="#" class="pure-menu-
link">Home</a></li>
```

```
            <li class="pure-menu-item"><a href="login.php" class="pure-
menu-link">Login</a></li>
        </ul>
        <?php
        }
        else
        {?>
        <ul class="pure-menu-list">
            <li class="pure-menu-item"><a href="#" class="pure-menu-
link">Home</a></li>
            <li class="pure-menu-item"><a href="logout.php" class="pure-
menu-link">Logout</a></li>
        </ul>
        <?php
        }
        ?>
    </div>
</div>

<div class="splash-container">
    <div class="splash">
        <h1 class="splash-head">My Inventory Website</h1>
        <?php
        if(!$log)
        {
        ?>
            <p class="splash-subhead">
                Login to access your inventory
            </p>
            <p>
                <a href="login.php" class="pure-button pure-button-
primary">Login</a>
            </p>
        <?php
        }
        else
        {?>
            <p class="splash-subhead">
                Logout here
            </p>
            <p>
                <a href="logout.php" class="pure-button pure-button-
primary">Logout</a>
                <br><BR><BR>
                <a href="viewinv.php" class="pure-button pure-button-
primary">View Inventory</a>
            </p>
        <?php
        }
        ?>
    </div>
</div>
</body>
</html>
```

Login.php

```
<?php
include('processlogin.php'); // Includes Login Script

// if logged in redirect
if(isset($_SESSION['login_user'])){
header("location: index.php");
}
?>

<!--  login form -->
<!DOCTYPE html>
<html>
<head>
<title>Login Form in PHP with Session</title>
<link rel="stylesheet" href="css/login.css">
</head>
<body>
<span onClick="window.location='index.php';" class="button" id="toggle-
login">Home</span>
<div id="login">
  <div id="triangle"></div>
  <h1>Log in</h1>
<form action="" method="post">
<label>UserName :</label>
<input id="name" name="username" placeholder="username" type="text">
<label>Password :</label>
<input id="password" name="password" placeholder="**********"
type="password">
<input name="submit" type="submit" value=" Login ">
<span  style="color:red"><?php echo $error; ?></span>
</form>
</div>
    <div class="login-help">
      <p>Forgot your password? <a href="password.php">Click here to reset
it</a>.</p>
    </div>
</body>
</html>
```

processLogin.php

```php
<?php
define( 'PARSE_SDK_DIR', './Parse/' );
// include Parse SDK autoloader
require './autoload.php';
// Add the "use" declarations where you'll be using the classes
use Parse\ParseClient;
use Parse\ParseObject;
use Parse\ParseQuery;
use Parse\ParseACL;
use Parse\ParsePush;
use Parse\ParseUser;
use Parse\ParseInstallation;
use Parse\ParseException;
use Parse\ParseAnalytics;
use Parse\ParseFile;
use Parse\ParseCloud;

$app_id = "saIQWlbzHC5G90hwMY0FdjnuN2dQInrRGMwuCMjy";
$rest_key = "PeyjycWPb7GyEZTSNpN2vOgevIQXR8wiOAjXmYCO";
$master_key = "Ja4mVcTfYllb5ovc9PkizltJtEwXvS2yvAdNZqcw";

    session_start(); // Starting Session
ParseClient::initialize( $app_id, $rest_key, $master_key );

    $error=''; // Variable To Store Error Message
    $user;
    if (isset($_POST['submit']))
    {
        if (empty($_POST['username']) || empty($_POST['password']))
        {
            $error = "Username or Password is invalid";
        }
        else
        {
            // Define $username and $password
            $username=$_POST['username'];
            $password=$_POST['password'];

            try {
              $user = ParseUser::logIn($username, $password);
              // Do stuff after successful login.
            } catch (ParseException $error) {
              $error = "No user";
            }
            // Establishing Connection with Server by passing server_name,
user_id and password as a parameter
            if (isset($user))
            {
                $_SESSION['theuser'] = $user;
                $_SESSION['login_user']=$user->get('username'); //
Initializing Session
                header("location: index.php"); // Redirecting To Other Page
            }
            else
            {
                $error = "Username or Password is invalid";
            }
        }
    }
?>
```

Password.php

```
<?php
include('processreset.php'); // Includes Login Script

if(isset($_SESSION['successReset'])){
}

?>
    <script src="http://code.jquery.com/jquery-1.9.1.js"></script>
    <script src="js/alertify.min.js"></script>

<link rel="stylesheet" href="css/login.css">
    <link rel="stylesheet" href="css/alertify.core.css" />

<span onClick="window.location='index.php';" class="button" id="toggle-
login">Home</span>

<!-- password reset form -->
<div id="login">
  <div id="triangle"></div>
  <h1>Reset Password</h1>
<form action="" method="post">
    <input type="email" placeholder="Email" name="emailaddy" />
    <input type="submit" name='submit' value="Send reset instructions" />
    <span  style="color:red"><?php echo $error; ?></span>
  </form>
</div>
```

processReset.php

```php
<?php
define( 'PARSE_SDK_DIR', './Parse/' );

// include Parse SDK autoloader
require './autoload.php';

// Add the "use" declarations where you'll be using the classes
use Parse\ParseClient;
use Parse\ParseObject;
use Parse\ParseQuery;
use Parse\ParseACL;
use Parse\ParsePush;
use Parse\ParseUser;
use Parse\ParseInstallation;
use Parse\ParseException;
use Parse\ParseAnalytics;
use Parse\ParseFile;
use Parse\ParseCloud;

$app_id = "saIQWlbzHC5G90hwMY0FdjnuN2dQInrRGMwuCMjy";
$rest_key = "PeyjycWPb7GyEZTSNpN2vOgevIQXR8wiOAjXmYCO";
$master_key = "Ja4mVcTfYllb5ovc9PkizltJtEwXvS2yvAdNZqcw";

ParseClient::initialize( $app_id, $rest_key, $master_key );

session_start(); // Starting Session
    $error=''; // Variable To Store Error Message
    $user;
    $success='';
    // check if submitted
    if (isset($_POST['submit']))
    {
        if (empty($_POST['emailaddy']))
        {
            $error = "Email is invalid";
        }
        else
        {
            // Define $username and $password
            $email=$_POST['emailaddy'];

            try {
              ParseUser::requestPasswordReset($email);
              header("location: index.php"); // Redirecting To Other Page
                // Password reset request was sent successfully
                $success = True;
            } catch (ParseException $ex) {
              // Password reset failed, check the exception message
              $error = "Email not found";
            }
        }
    }
?>
```

viewInv.php

```php
<?php

include('processlogin.php'); // Includes Login Script

// if user logged in
if(isset($_SESSION['login_user'])){
$log = True;
$user = $_SESSION['theuser'];
}

// else redirect
else{
    header("location: index.php");
$log = False;
}

// include Parse SDK autoloader
require 'autoload.php';

// Add the "use" declarations where you'll be using the classes
use Parse\ParseClient;
use Parse\ParseObject;
use Parse\ParseQuery;
use Parse\ParseACL;
use Parse\ParsePush;
use Parse\ParseUser;
use Parse\ParseInstallation;
use Parse\ParseException;
use Parse\ParseAnalytics;
use Parse\ParseFile;
use Parse\ParseCloud;

// Init parse: app_id, rest_key, master_key
//ParseClient::initialize('SCg17O8f0uFlm1DK3BTraClFMsTNiGUa8n2SCTAL',
'erWKvEaj0E8zG7Z6XVrnYjJXRFY0kV7uFZglEqcK',
'Ff1K5xELw3a8Wmh2iZOkcHKnaodGRdaRZHmn4Vd4');
$app_id = "saIQWlbzHC5G90hwMY0FdjnuN2dQInrRGMwuCMjy";
$rest_key = "PeyjycWPb7GyEZTSNpN2vOgevIQXR8wiOAjXmYCO";
$master_key = "Ja4mVcTfYllb5ovc9PkizltJtEwXvS2yvAdNZqcw";


ParseClient::initialize( $app_id, $rest_key, $master_key );

// get current user
$currentUser = ParseUser::getCurrentUser();

// query
$query = new ParseQuery("Inventory");
$query->ascending("title");
$results = $query->find();

// current user id
$currentUserID = $currentUser->getObjectId();
?>

<!-- html form -->

<!doctype html>
<html lang="en">
<head>
```

```
    <meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta name="description" content="A layout example that shows off a
responsive product landing page.">

<title>Home Page &ndash;</title>

<link rel="stylesheet" href="http://yui.yahooapis.com/pure/0.6.0/pure-
min.css">


<!--[if lte IE 8]>

    <link rel="stylesheet" href="http://yui.yahooapis.com/pure/0.6.0/grids-
responsive-old-ie-min.css">

<![endif]-->
<!--[if gt IE 8]><!-->

    <link rel="stylesheet" href="http://yui.yahooapis.com/pure/0.6.0/grids-
responsive-min.css">

<!--<![endif]-->

<link rel="stylesheet" href="http://netdna.bootstrapcdn.com/font-
awesome/4.0.3/css/font-awesome.css">


    <!--[if lte IE 8]>
        <link rel="stylesheet" href="css/layouts/marketing-old-ie.css">
    <![endif]-->
    <!--[if gt IE 8]><!-->
        <link rel="stylesheet" href="css/layouts/marketing.css">
    <!--<![endif]-->



    <style>
li.one a {
  text-decoration: none;
  color: #000;
  display: block;
  width: 1800px;
 font-weight: bold;
 line-height: 0.8em;

  -webkit-transition: font-size 0.3s ease, background-color 0.3s ease;
  -moz-transition: font-size 0.3s ease, background-color 0.3s ease;
  -o-transition: font-size 0.3s ease, background-color 0.3s ease;
  -ms-transition: font-size 0.3s ease, background-color 0.3s ease;
  transition: font-size 0.3s ease, background-color 0.3s ease;
}

li.one a:hover {
  font-size: 25px;
  background: #1f8dd6;
  color:white;

}

.numbers{
```

```
      color:white;
      margin-left: 25px;
}
    </style>
</head>
<body>

<!-- header  -->
<div class="header">
    <div class="home-menu pure-menu pure-menu-horizontal pure-menu-fixed">
        <a class="pure-menu-heading" href="">My Inventory</a>

                <ul class="pure-menu-list">

            <li class="pure-menu-item"><a href="index.php" class="pure-
menu-link">Home</a></li>
            <li class="pure-menu-item"><a href="logout.php" class="pure-
menu-link">Logout</a></li>
        </ul>
            <select>
                <?php foreach($results as $key) { ?>
                <option value="<?php echo $key->get('title') ?>"><?php echo
$key->get('category') ?></option>
                <?php }?>
            </select>
    </div>
</div>

<div class="splash-container">

        <BR>

        <?php

        // display pagination

        ini_set('display_errors','On');
        error_reporting(E_ALL);

        // Include the pagination class
        include 'pagination.class.php';

        for ($i = 0; $i < count($results); $i++) {
              $objTitle[] = $results[$i]->get('title');
        }

        // If we have an array with items
        if (count($results)) {
          // Create the pagination object
          $pagination = new pagination($results, (isset($_GET['page']) ?
$_GET['page'] : 1), 14);
          // Decide if the first and last links should show
          $pagination->setShowFirstAndLast(false);
          // You can overwrite the default seperator
          $pagination->setMainSeperator(' | ');
          // Parse through the pagination class
          $productPages = $pagination->getResults();
          // If we have items
          if (count($productPages) != 0) {
            // Create the page numbers
```

```
            echo $pageNumbers = '<div class="numbers">'.$pagination-
>getLinks($_GET).'</div>';
            // Loop through all the items in the array
            ?><BR><BR><BR><?php

            foreach ($productPages as $productArray) {
              // Show the information about the item
              ?>
<!--hr-->
            <form>
            <div >

            <ul style="color:white" >
            <?php
            // get current users items
            if($currentUserID == $productArray->get('author')-
>getObjectId())
            {
            ?>
            <br>
                <li class = "one"><a class="abc" href =
"itemDetails.php?id=<?php echo $productArray->getObjectId(); ?>"><?php echo
$productArray->get('title'); ?></a></li></hr>
            <?php
            }
            ?>
            </ul>

            </div>

          </form>

          <?php
            }
            ?> <!--hr--><?php
            // print out the page numbers beneath the results
            echo "\x20\x20\x20";
            echo $pageNumbers;
          }

        }
      ?>
</div>
</body>
</html>
```

itemDetails.php

```php
<?php
define( 'PARSE_SDK_DIR', './Parse/' );

require 'autoload.php'; // Includes Login Script

// Add the "use" declarations where you'll be using the classes
use Parse\ParseClient;
use Parse\ParseObject;
use Parse\ParseQuery;
use Parse\ParseACL;
use Parse\ParsePush;
use Parse\ParseUser;
use Parse\ParseInstallation;
use Parse\ParseException;
use Parse\ParseAnalytics;
use Parse\ParseFile;
use Parse\ParseCloud;

// Init parse: app_id, rest_key, master_key
$app_id = "saIQWlbzHC5G90hwMY0FdjnuN2dQInrRGMwuCMjy";
$rest_key = "PeyjycWPb7GyEZTSNpN2vOgevIQXR8wiOAjXmYCO";
$master_key = "Ja4mVcTfYllb5ovc9PkizltJtEwXvS2yvAdNZqcw";

ParseClient::initialize( $app_id, $rest_key, $master_key );

$theId = $_GET["id"];

// the query
$query = new ParseQuery("Inventory");
$query->equalTo("objectId", $theId);
$query->ascending("title");
$object = $query->first();

?>

<!-- item details form -->
<!DOCTYPE html>
<html>
<head>
<script type='text/javascript'>
function confirmDelete()
{
   return confirm("Are you sure you want to delete this?");
}
</script>

<title>Item Details</title>
<link rel="stylesheet" href="css/login.css">

<style>
#login{
  width:600px;
}

textarea {
   resize: none;
  width:92%;
  background:#fff;
  margin-bottom:0%;
  border:1px solid #ccc;
```

```
  padding:4%;
  font-family:'Open Sans',sans-serif;
  font-size:95%;
  color:#555;
}

input[type="button"]#delete {

background:#1f8dd6;

}
input[type="submit"]#delete:hover{
  background:#FF0000;
}
</style>

<!-- display details into text boxes -->
</head>
<body>
<span onClick="window.location='index.php';" class="button" id="toggle-
login">Home</span>
<div id="login">
  <div id="triangle"></div>
  <h1>Item Details</h1>
<form action="" method="post">
<label>Title :</label>
<input name="titleBox" type="text" value="<?php echo( htmlspecialchars(
$object->get('title')) ); ?>" />
<label>Description :</label>
<textarea name="descBox" cols = "67" rows="2"><?php echo( htmlspecialchars(
$object->get('description')) ); ?></textarea>
<label>Category :</label>
<input name="cateBox" type="text" value="<?php echo( htmlspecialchars(
$object->get('category')) ); ?>" />
<label>Author / Artist :</label>
<input name="authorBox" type="text" value="<?php echo( htmlspecialchars(
$object->get('category')) ); ?>" />
<center>
<img src="<?php echo ($object->get('image'))?>" width="150" height="200"
alt="Nba"></center>

<input name="update" type = "submit" value = "Update">
<hr><br>
<input name="delete" id="delete" type = "submit" value = "Delete"
onClick="return confirmDelete()">

</form>
</div>
    <div class="login-help">
      <p><a href="viewinv.php">Go back to inventory</a>.</p>
    </div>
</body>
</html>


<?php
// update item
if (isset($_REQUEST['update'])) {
$test = $_POST['titleBox'];
$test2 = $_POST['descBox'];
$test3 = $_POST['cateBox'];
```

```php
$object->set("title",$test);
$object->set("description",$test2);
$object->set("category",$test3);

$object->save();

header("Location: viewinv.php");
}
?>

<?php
// delete item
if (isset($_REQUEST['delete'])) {

$object->destroy();
echo "<script type='text/javascript'>alert('Deleted');</script>";
header("Location: viewinv.php");
}

?>
```