Criterion E: Product development

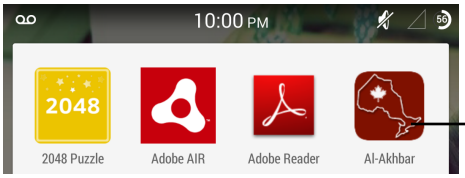
Complex techniques used to create the final product

The complex Java and XML features used to build the structure of the Al-Akhbar Android application are listed below:

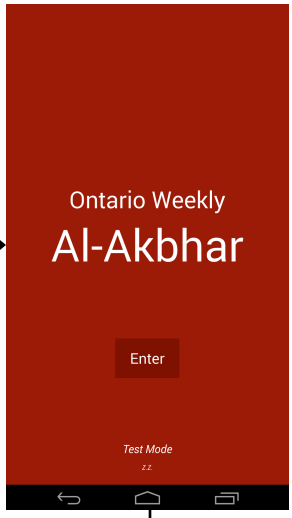
- Client-server based application through HTTP connection.....(4 – 7)
    - Java activities and objects
    - AsyncTask download thread
    - Instantaneous news updates
- Offline application service by reading/writing files in internal storage .....(8 – 12)
    - If statement decision making: Internet connection management: 3G/WiFi
    - Reading/writing device storage permission
    - Buffer streams to write bytes of data
    - Updating app content with JSON parsing - advanced arrays and lists
- Online database to manage news items .....(13 – 15)
    - SQL queries/user interface to update news records
    - Godaddy hosted through personal domain
    - JSON database creation
    - Hashmap key-value pairings to categorize news items
- Use of XML and Java design to create a user-friendly layout.....(16 – 23)
    - Special color scheme and app logo design
    - HTML tags parsing
    - Tabbed activities
    - Custom ListView adapter to customize news items
    - Graphics manipulation, Photoshop
    - Dialog boxes
    - XML based graphical user interface (GUI)

All of the code relating to the extra features of the application is snapshotted at the end of Criterion E. NOTE: The code to develop such extra features was taken from online sources (mentioned in Criterion D), and then manipulated to fit the specifications of my application. Only the structure and concept of the code was copied, the code was completely rewritten by me.

Overall structure of the Al-Akhbar Android application

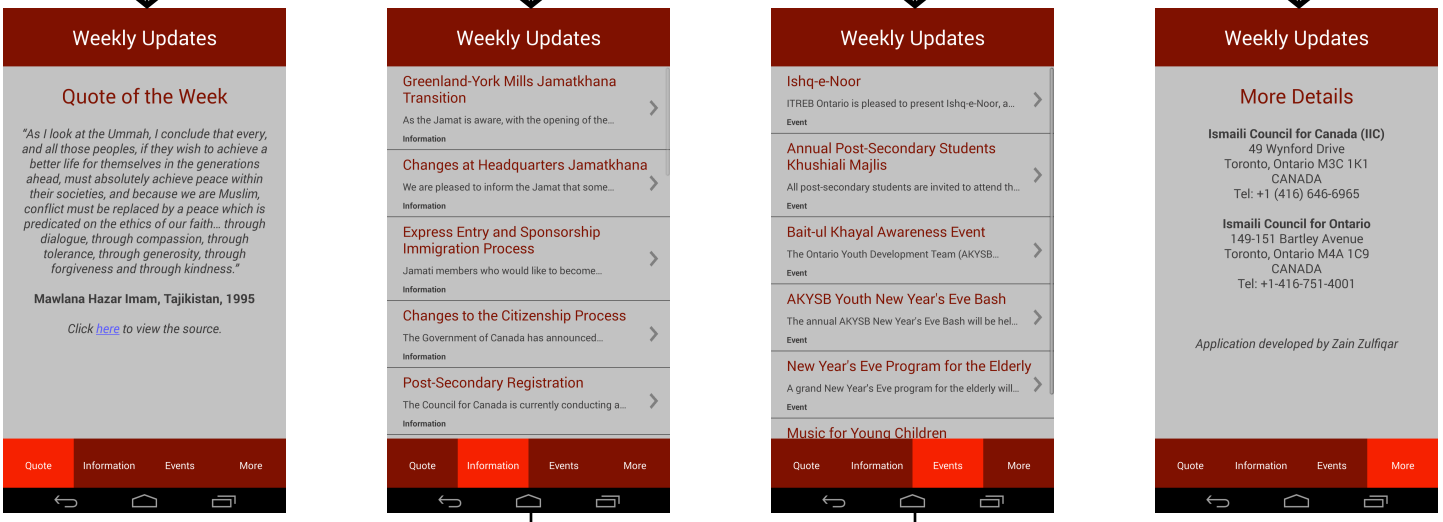


1. App icon launcher to start the application

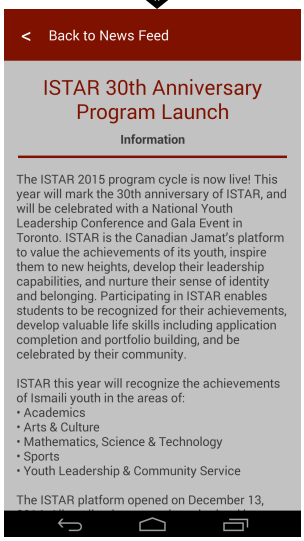


The Android application is composed of 7 main activities: app icon launcher, home screen, weekly quote screen, information screen, events screen, and more screen. These screens are shown below. The full Java and XML code to develop the Android application is presented at the end.

2. Home screen with an enter button to start download process

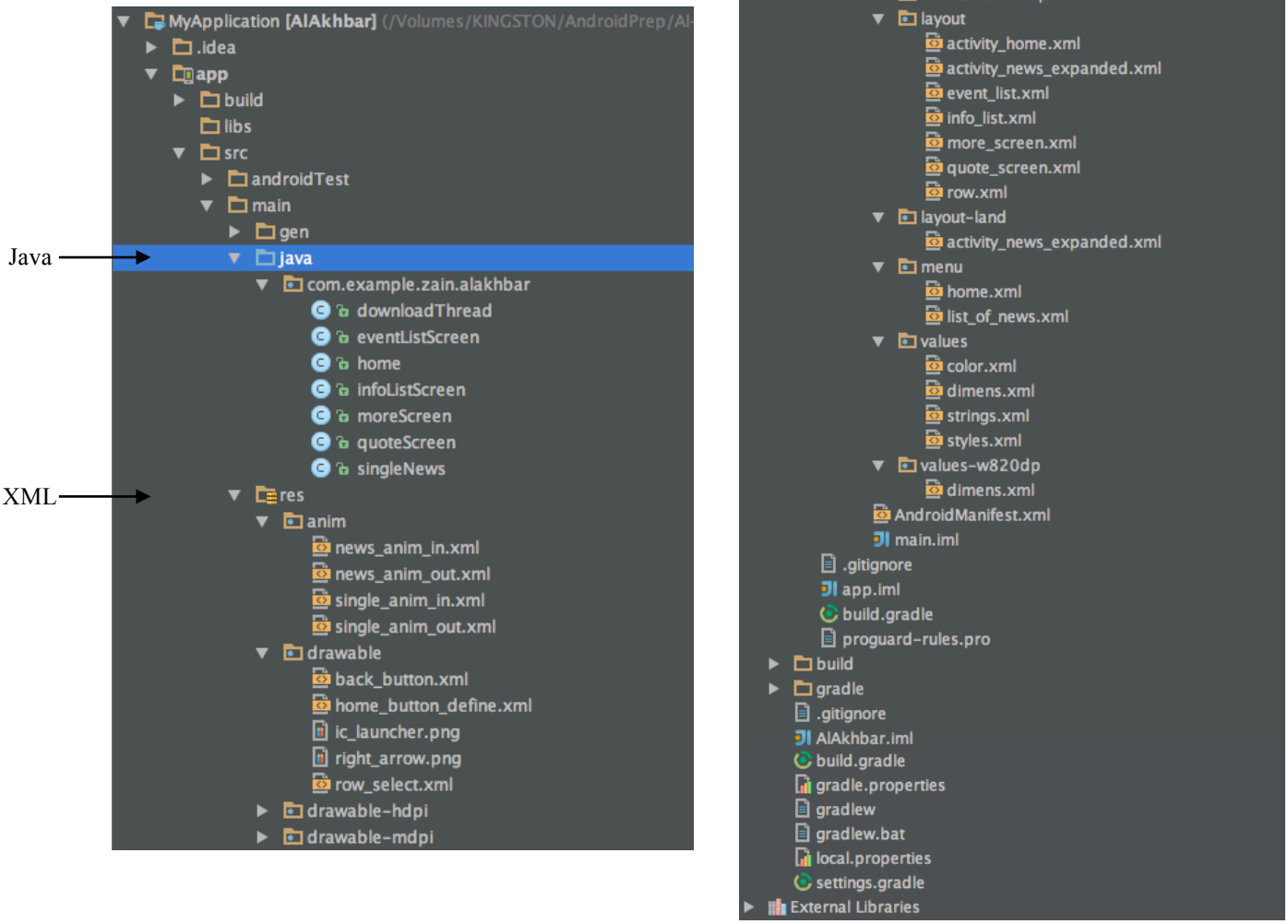


3. Tabbed screens containing the organized news items in lists.



4. Individual news screen to provide more details for a news item the user clicks on

Android Studio IDE was used to develop the application. It created its own grade-based compiling files for the Android application. The expanded folders contain all sections where my Java or XML code exists.



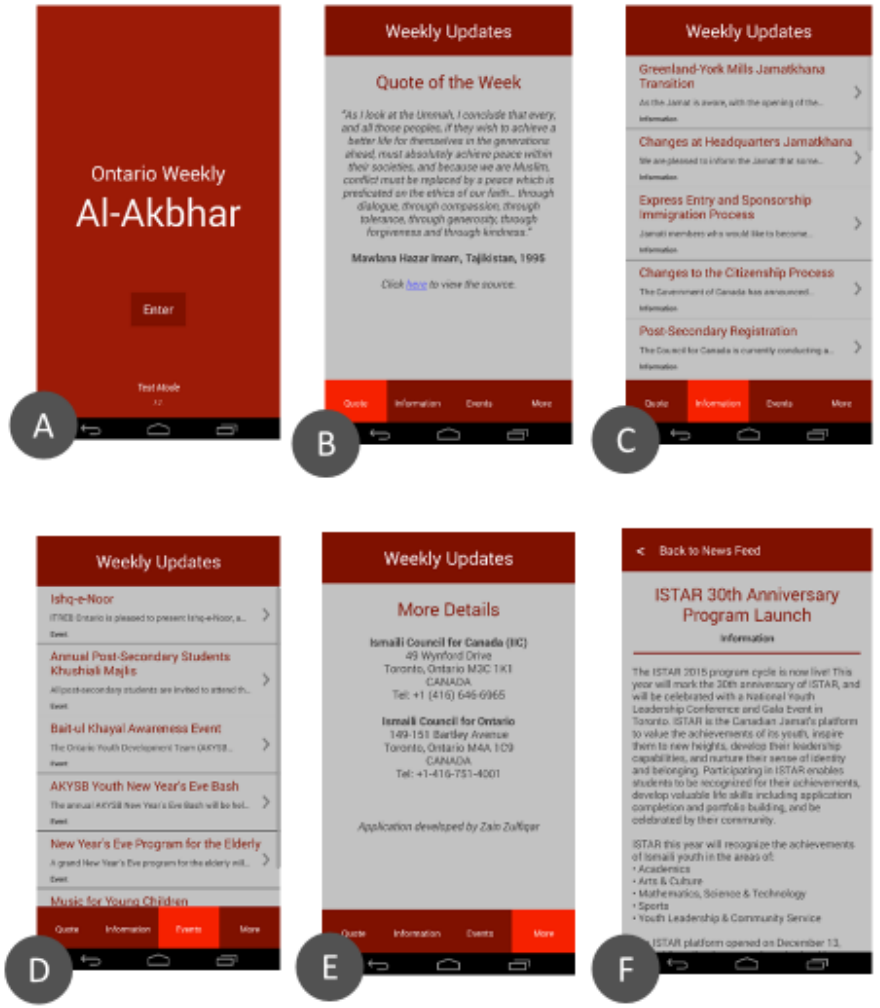
Client-server based application through HTTP connection

All screens have their own Android OS activity and processes to allow independent memory allocation. This means that if one part of the application fails due to low memory, the background screens that are not being viewed by the user can be closed. This ensures the application has a smooth performance, and does not clog up memory in the user's device[1].

Every single screen that the user can see has its own activity initialized in the AndroidManifest.XML file.

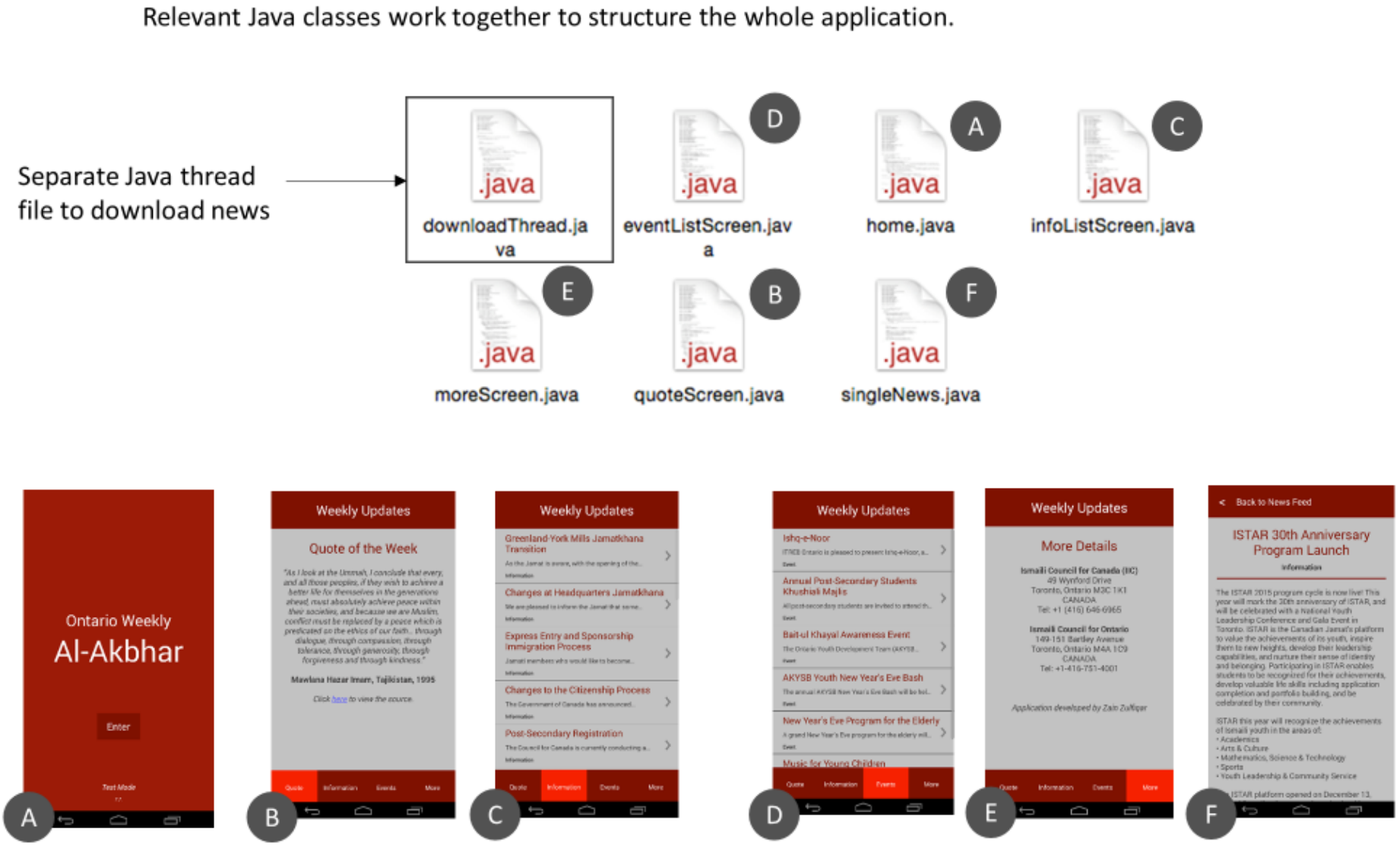
```
<activity
    android:name=".home"
    android:label="@string/app_name"
    android:screenOrientation="portrait">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
<activity
    android:name=".quoteScreen"
    android:label="@string/title_activity_list_of_news"
    android:screenOrientation="portrait">
</activity>
<activity
    android:name=".infoListScreen"
    android:label="@string/title_activity_list_of_news"
    android:screenOrientation="portrait">
</activity>
<activity
    android:name=".eventListScreen"
    android:label="@string/title_activity_list_of_news"
    android:screenOrientation="portrait">
</activity>
<activity
    android:name=".moreScreen"
    android:label="@string/title_activity_list_of_news"
    android:screenOrientation="portrait">
</activity>
<activity
    android:name=".singleNews"
    android:label="@string/title_activity_list_of_news"
    android:screenOrientation="portrait">
</activity>
```

A B C D E F



[1] Criterion B, Specification 2: Application must be user friendly and work smoothly with the user.

All code files are organized in their appropriate folders. This allows anyone from the IIC to customize the application without any confusion regarding the location of the source code files. The files are appropriately named by their function so the IIC understands the application structure. Classes help with data abstraction. The IIC does not need to understand the inner-details of the Android activity and can have an abstract understanding of the class functions. Additionally, modularity of classes allows for the subdividing of code between different committees in the IIC to independently work on different sections of the application[2].



[2] Criterion B, Specification 1: Create an application/system to display weekly news updates, event information, and Quote of the Week from Ismaili Institution Canada (IIC).

Permissions to access the Internet/storage settings of the device are needed to establish a client-server system. The application informs the user of the changes that the application will make to their device. This gives the user an option to "opt-out" of the system if they do not feel comfortable, making sure the user is aware of the application's operation and trust this new system the IIC is trying to implement[3].

Establishing reading/writing device storage permissions

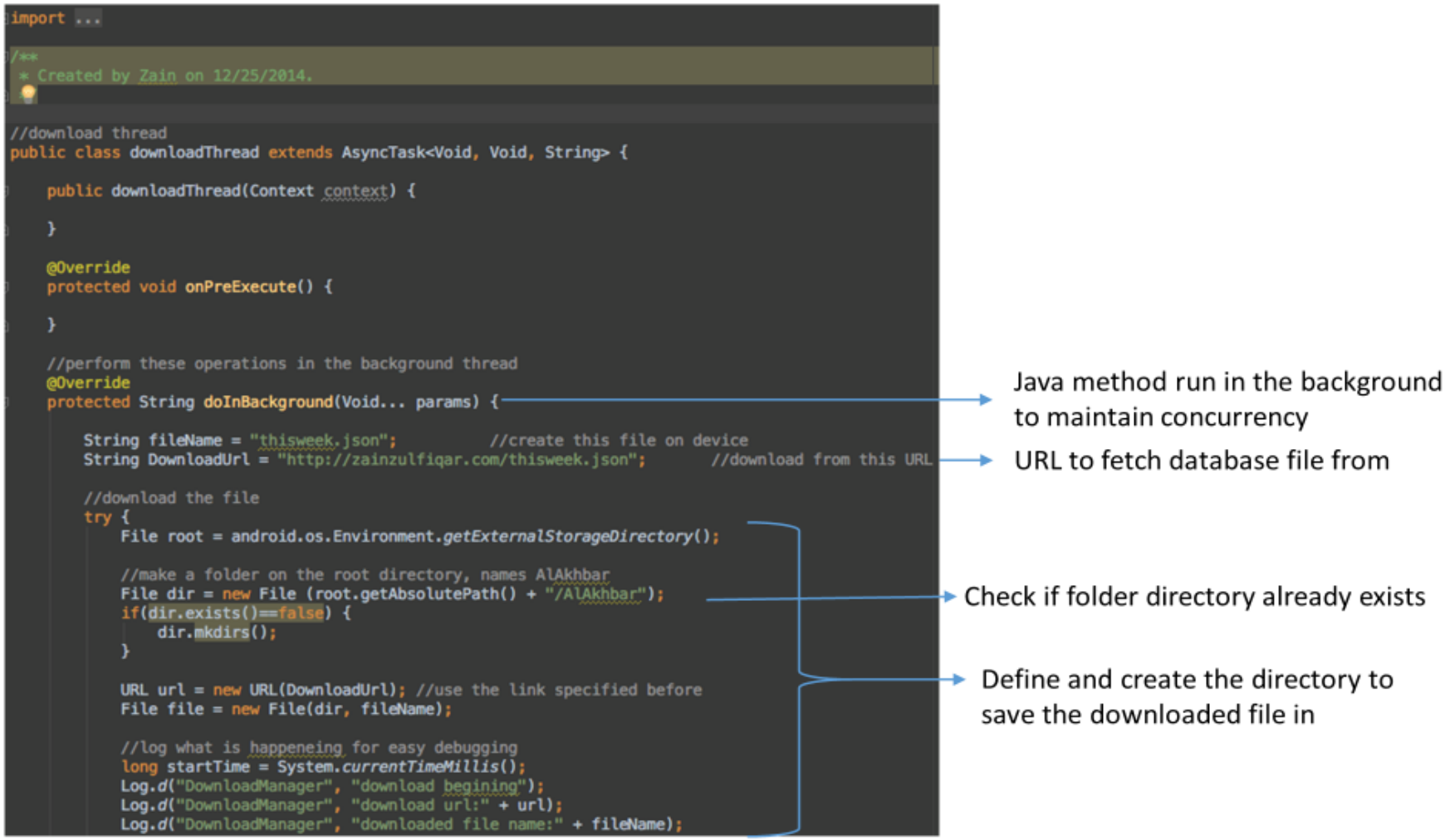
```
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
```

The following permissions are defined for the application in order:
- Check current WiFi state
- Access the Internet
- Check Internet connection state
- Write to the external storage
- Read from the external storage
- Read the phone's memory and processing state

[3] Criterion B, Specification 2: Application must be user friendly and work smoothly with the user.

A thread is launched to download the news items from the host when the “Enter” button is pressed. The thread establishes a back-end connection with the IIC database, while the user continues to scroll through the present items in the application, so that the update process is parallel and does not interrupt application usage[4]. This concurrent programming makes the interface more fluid, and user-friendly[5] as there is no delay in the loading of items.

downloadThread.java file showing the background thread process when downloading news



[4] "AsyncTask." AsyncTask | Android Developers. Accessed January 17, 2015.http://developer.android.com/reference/android/os/AsyncTask.html.
[5] Criterion B, Specification 2: Application must be user friendly and work smoothly with the user.

Offline application service by reading/writing files in internal storage

Bufferstream is used to save the downloaded database content onto storage for offline usage. Users can check for reminders/important dates and time while away from home, on the go, making the application more convenient. Reading operations with files are slow. To minimize the operations from the online database, a buffer stream is used to store the database file into RAM and save time when processing the file onto the local storage[6].

downloadThread.java file downloading the information and saving the bytes into a file

```
// Open a connection to that URL.
 URLConnection ucon = url.openConnection();

//Define InputStreams to read from the URLConnection.
 InputStream is = ucon.getInputStream();
 BufferedInputStream bis = new BufferedInputStream(is);

//Read bytes to the Buffer until there is nothing more to read(-1).
 ByteArrayBuffer baf = new ByteArrayBuffer(5000);
 int current = 0;
 while ((current = bis.read()) != -1) {
     baf.append((byte) current);
 }

// Convert the Bytes read to a String.
 FileOutputStream fos = new FileOutputStream(file);
 fos.write(baf.toByteArray());
 fos.flush();
 fos.close();
```

Establish a buffer stream to write bytes of data into a file

Save file into a readable String so that th contents can be understood by Java

[6] "BufferedInputStream." BufferedInputStream | Android Developers. Accessed January 18, 2015. http://developer.android.com/reference/java/io/BufferedInputStream.html

Internet connection is checked to ensure the database can be downloaded. Based on this, the application either launches the Internet settings, or the offline system. Nested if statements help to determine the connection state by returning Boolean values.[7] This system prevents any confusion of why content may not be updating properly or why news items don't appear.

In the home.java file, check to see if internet connection is available, to see if download thread can be launched

```
//check to see if connected to network
public boolean isConnected(Context context) {

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

    if (netinfo != null && netinfo.isConnectedOrConnecting()) {
        android.net.NetworkInfo wifi = cm.getNetworkInfo(ConnectivityManager.TYPE_WIFI);
        android.net.NetworkInfo mobile = cm.getNetworkInfo(ConnectivityManager.TYPE_MOBILE);

        if((mobile != null && mobile.isConnectedOrConnecting()) || (wifi != null && wifi.isConnectedOrConnecting())) return true;
        else return false;
    } else
        return false;
}
```

Check to see if local WiFi connection

Or if it is it a 3G/mobile connection

Return **true** if any of these conditions are met/in progress (internet connection available), otherwise return **false**

[7] "Detect Whether There Is an Internet Connection Available on Android." Stack Overflow. Accessed January 17, 2015. http://stackoverflow.com/questions/4238921/detect-whether-there-is-an-internet-connection-available-on-android.

During first time application use, there is no database file in storage. Each scenario has its own conditional if statement to ensure application runs well all the time no matter what the connection state. This makes the application easier to use, as the user does not have to deal with Internet error handling when fetching the news.

Check to see if Internet connection is needed, if database file already exists, and if offline content is available

```
File root = android.os.Environment.getExternalStorageDirectory();
File file = new File (root.getAbsolutePath() + "/AlAkhbar/thisweek.json");
if(!file.exists() && !isConnected(home.this)) {          //if file is not there,
    System.out.println(""+file.exists());
    buildDialog(home.this).show();
}

else if(file.exists() && !isConnected(home.this)) {                  //if file
    Intent i = new Intent(view.getContext(), quoteScreen.class);
    startActivity(i);
}

else if(isConnected(home.this)) {                    //if file is there, and co
    downloadNews.execute();
    Intent i = new Intent(view.getContext(), quoteScreen.class);     //setup th
    startActivity(i);       //start the new screen
}
```

If file and internet connection does not exist (first time app use), bring dialog box to redirect user to the internet settings page

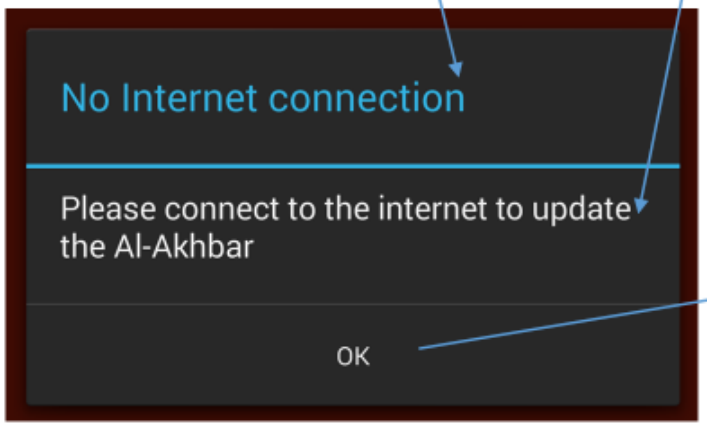
If file exists, but no internet connection, use the file and display the content on a new screen (offline connection)

If internet exists (even if file does or doesn't exist), launch the download thread to update database file onto user's storage, and show the content.

A dialog box informs the user if connection needed. The reason for the connection is explained, and the user is taken to settings. Dialog boxes help guide the user in the direction of the application process. Users cannot perform any other action that may crash the application. This ensures the application runs as it was designed, and that the user avoids accidental problems[8].

If no internet connection, and no database file in local storage (first time use), spawn dialog box for internet connection

```
//spawn an alert dialog to inform the user that there is no Internet connection
public AlertDialog.Builder buildDialog(Context c) {

    AlertDialog.Builder builder = new AlertDialog.Builder(c, AlertDialog.THEME_HOLO_DARK);
    builder.setTitle("No Internet connection");
    builder.setMessage("Please connect to the internet to update the Al-Akhbar for first-time use.");

    builder.setPositiveButton("OK", (dialog, which) -> {            //on clicking the OK button, do thi
            dialog.dismiss();
            startActivity(new Intent(Settings.ACTION_WIFI_SETTINGS));       //launch the Android system
    });

    return builder;
}
```



Launch internet settings once the OK button is clicked

[8] Criterion B, Specification 2: Application must be user friendly and work smoothly with the user.

The downloaded JSON file is filtered to display the relevant items in each tab[9]. This organization helps the user see relevant details in the one location. The user is able to see information that relates to their interests or needs, instead of getting overwhelmed with unnecessary details. News items are stored in array hash-map key-value pairs to package details together as one object, efficiently passing information throughout the application. Packaging items together ensures all related content is delivered in one compressed process to reduce CPU power.

Update news content from news database file after parsing the JSON formatted file in local storage – eventListScreen.java, moreScreen.java, and infoListScreen.java

```
//read the downloaded JSON file from online SQL server, and parse it into a string format
try {
    File yourFile = new File(Environment.getExternalStorageDirectory(), "AlAkhbar/thisweek.json");
    FileInputStream stream = new FileInputStream(yourFile);
    try {
        FileChannel fc = stream.getChannel();
        MappedByteBuffer bb = fc.map(FileChannel.MapMode.READ_ONLY, 0, fc.size());

        jsonStr = Charset.defaultCharset().decode(bb).toString();
    }
    finally {
        stream.close();
    }

    //create a JSON object, to parse file
    JSONObject jsonObj = new JSONObject(jsonStr);

    // specify the table name (table name is an array, that contains all the element of the table) of the JSON file
    news = jsonObj.getJSONArray(NEWSTHISWEEK);

    // looping through all the elements in the table
    for (int i = 0; i < news.length(); i++) {
        JSONObject c = news.getJSONObject(i);

        String title = c.getString(TITLE);
        String body = c.getString(BODY);
        String subtitle = c.getString(SUBTITLE);
        String newsType = c.getString(NEWSTYPE);

        //use if statement to filter what content we want to display in this activity
        if (newsType.equals("e")) {
            newsType = "Event";

            // tmp hashmap for single contact
            newsItems = new HashMap<String, String>();

            // adding each child node to HashMap key => value
            newsItems.put(TITLE, title);
            newsItems.put(BODY, body);
            newsItems.put(SUBTITLE, subtitle);
            newsItems.put(NEWSTYPE, newsType);

            // adding news to news list
            newsList.add(newsItems);
```

Read news database file from storage and stream it into a String variable

Create JSON object in Java to interpret JSON formatting of database file in the String variable

Use a for loop to go through **all** of the news elements in the database file, until no more records to read

Special string variable to store each element of the news from the JSON database file.

Filter through to take only the news relevant to the tab user is on. If user on events tab, only news with “e” tag taken, if user on information tab, only news with “i” tag taken.

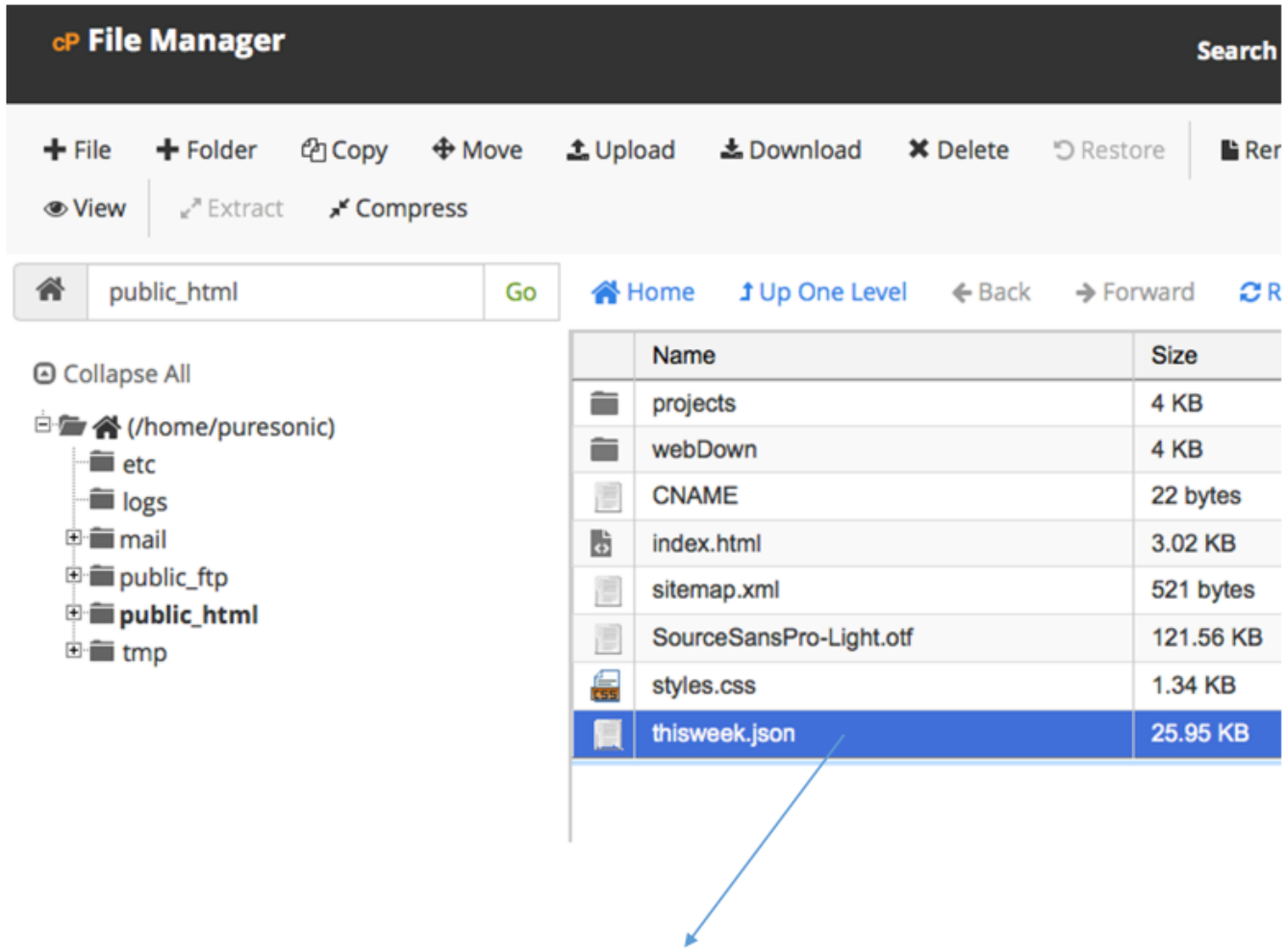
Combine the news elements to create one news element to add into the list presented to user.

[9] "How to Build a Mobile App with an App Engine Backend (Tutorial)." Google Developers. Accessed January 18, 2015.

Online database to manage news items

Database file is hosted on the GoDaddy server 24/7. This means users can access news item updates at any time that is convenient for them. All users are updated instantaneously and simultaneously[10].

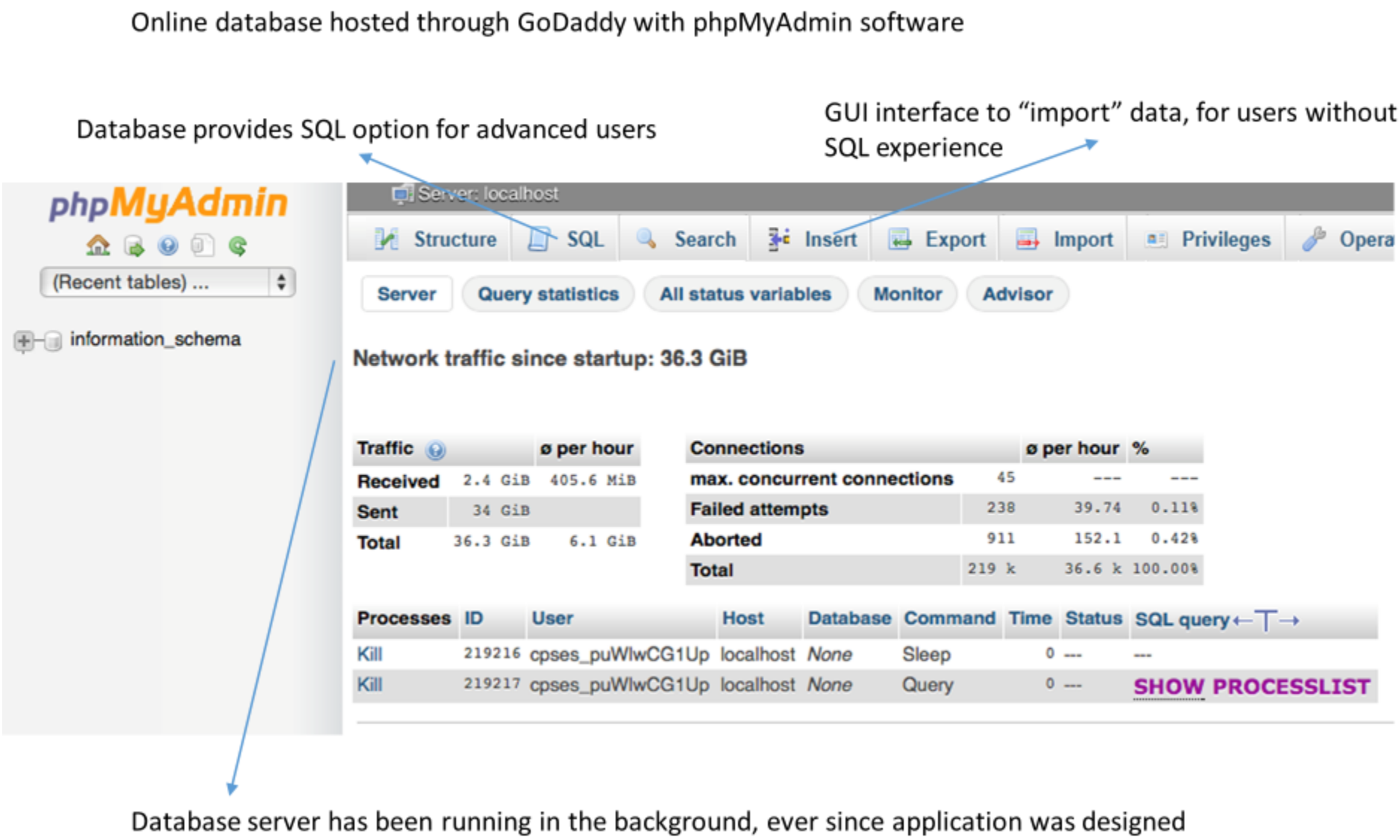
Online database file hosted through GoDaddy domain servers – zainzulfiqar.com/thisweek.json



Database file exported to local website storage, hosted 24/7 on GoDaddy

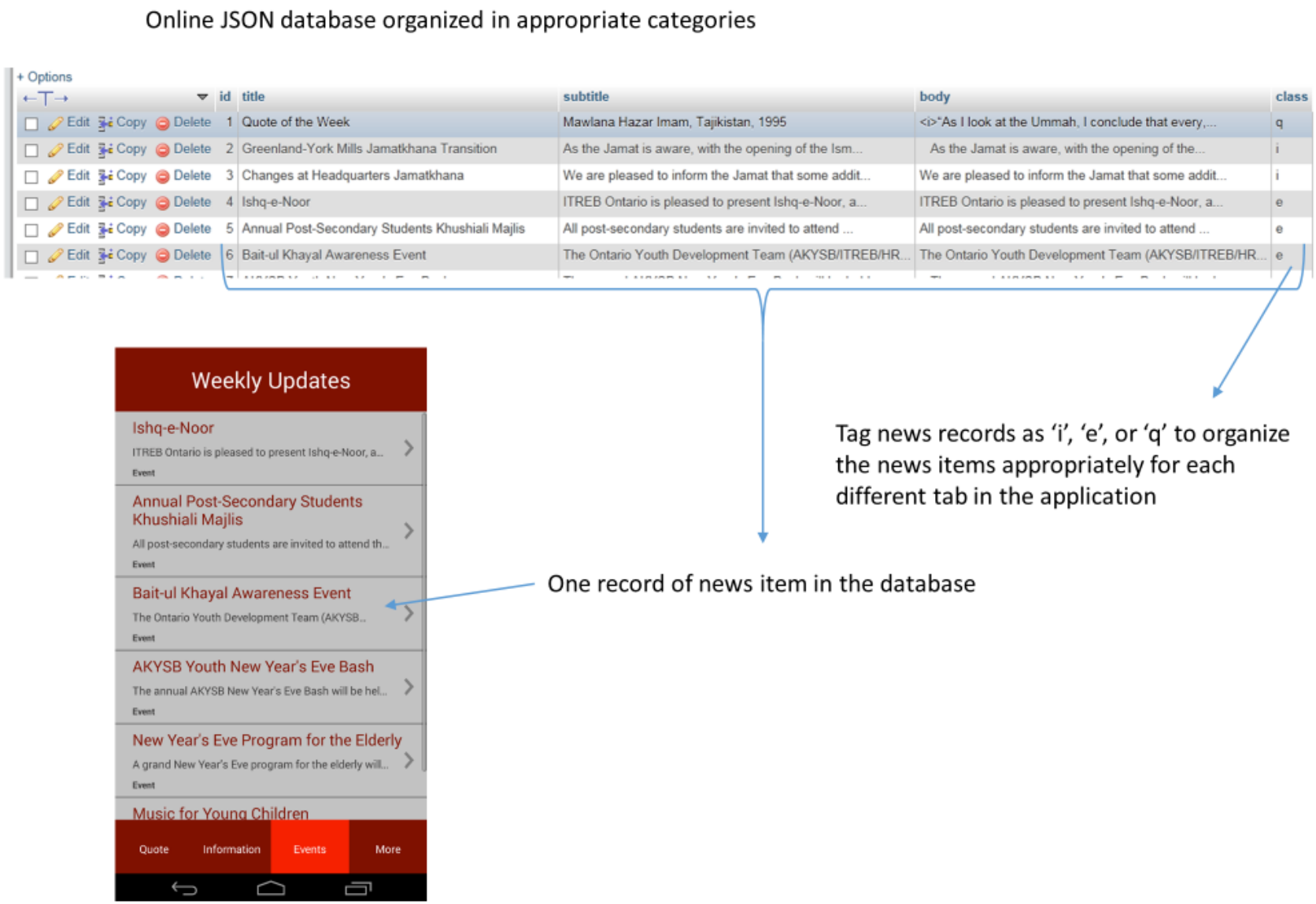
[10] Criterion B, Specification 5: Create a server-end application to allow IIC committee members to input information into the database and keep the users updated.

The online database uses SQL or a visual GUI to input news records. The system is user-friendly, allowing the IIC to update their users in a fast and easy process. IIC users of any skill level can update the database, allowing flexibility when dividing the database entries within the committee.[11].



[11] Criterion B, Specification 5: Create a server-end application to allow IIC committee members to input information into the database and keep the users updated.

"Class" to organize news[12].

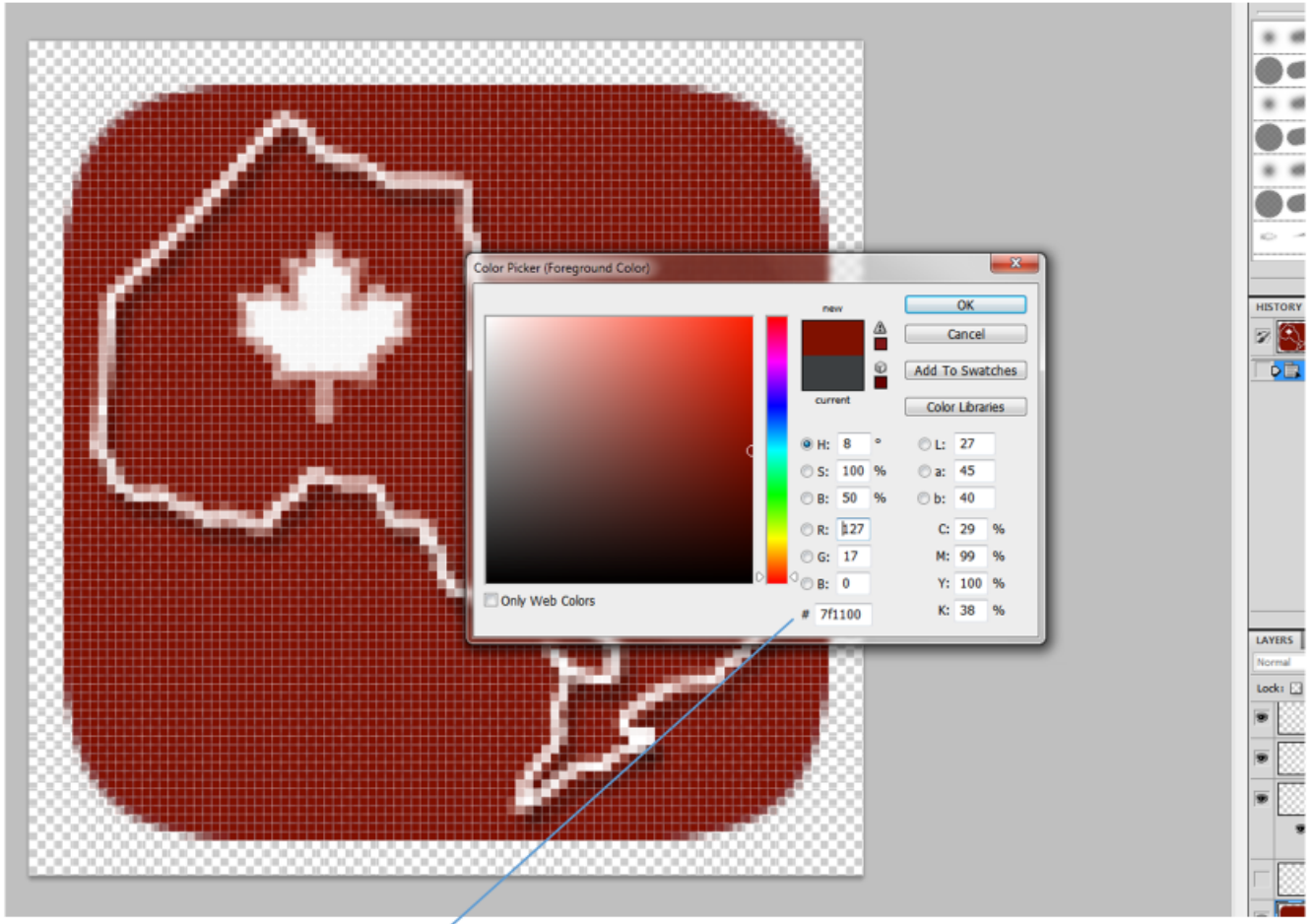


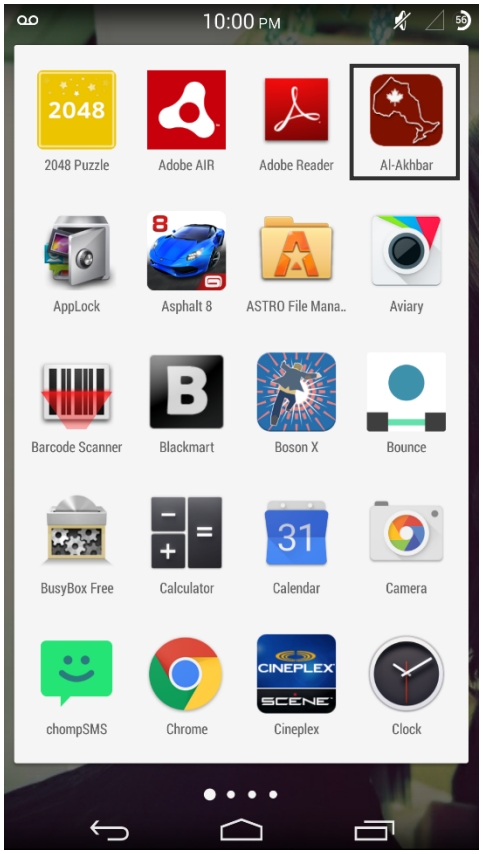
[12] Criterion B, Specification 6: Database entries are user friendly for the IIC, and organize content appropriately and automatically.

Use of XML and Java design to create a user-friendly interface

An application logo, designed in Photoshop, needed to reflect the application's purpose: a news service for Ontario Ismailis. The outline of the Ontario map was placed. The background was chosen to be red, so that the color scheme matches the IIC's. Also, the red background helps to quickly find the app icon in the white app drawer[13].

Custom app logo/icon designed to represent the Android application



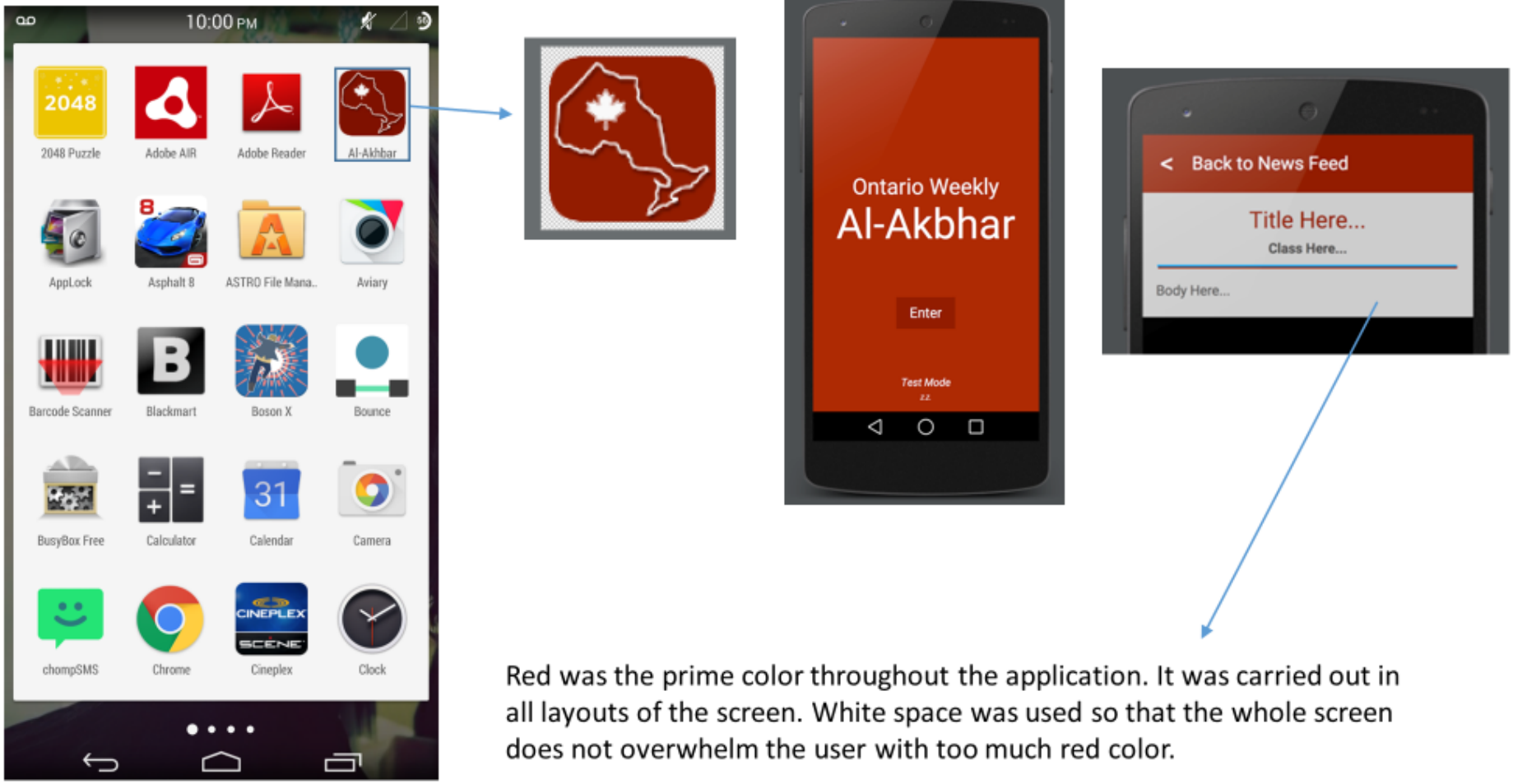


Specific color scheme of red was matched to the IIC's theme, design was implemented in Photoshop

[13] Criterion B, Specification 3: Application should be visually appealing, with a logo.

The IIC uses red in their website, emails, and mail packages. Ismailis in Ontario associate red with the IIC. To represent the same committee, a red color scheme was used. Implementing the graphics used by the IIC into the application became easy, without worrying about color matching or redesigning.[14]

Red color scheme carried out through the application



Red was the prime color throughout the application. It was carried out in all layouts of the screen. White space was used so that the whole screen does not overwhelm the user with too much red color.

[14] Criterion B, Specification 3: Application should be visually appealing, with a logo.

The IIC can manually place HTML tags into news records to customize specific sections of the items. This helps to guide the user's eye to emphasize important deadlines, highlight hyperlinks, and uniquely format each item for effective information density[15].

HTML tags are inserted into the database file with the news content so that each news can be individually customized

```
//get data
Intent i = getIntent();
String[] newsExpandedValues = i.getStringArrayExtra("newsData");

//update
TextView title = (TextView) findViewById(R.id.singleTitle);
TextView newsType = (TextView) findViewById(R.id.singleClass);
TextView body = (TextView) findViewById(R.id.singleBody);
title.setText(newsExpandedValues[0]);
newsType.setText(newsExpandedValues[1]);

// body setup
body.setText(Html.fromHtml(newsExpandedValues[2]));
body.setMovementMethod(LinkMovementMethod.getInstance());
```

Java HTML parser, converting the tags into visual styles when user views the news.

HTML tags such as
 to create line breaks, <b> to bold specific sections of text

```
body.setText(Html.fromHtml("<b>Ismaili Council for Canada (IIC)</b><br>" +
        "49 Wynford Drive<br>" +
        "Toronto, Ontario M3C 1K1<br>" +
        "CANADA<br>" +
        "Tel: +1 (416) 646-6965<br><br>" +
        "<b>Ismaili Council for Ontario</b><br>" +
        "149-151 Bartley Avenue<br>" +
        "Toronto, Ontario M4A 1C9<br>" +
        "CANADA<br>" +
        "Tel: +1-416-751-4001<br><br><br><br>" +
        "<i>Application developed by Zain Zulfiqar</i>"));
```

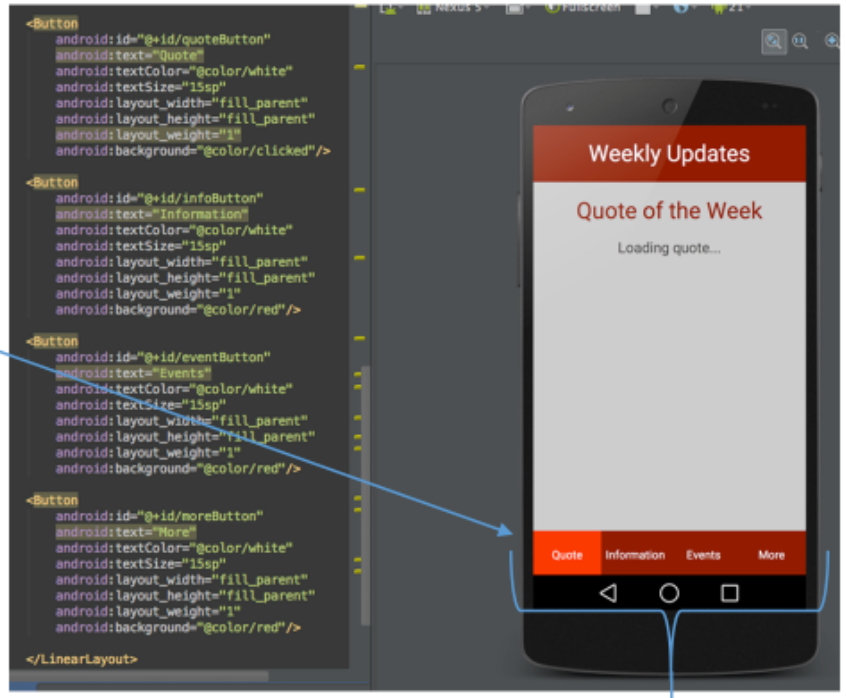
[15] Criterion B, Specification 3 and 4: Application should be visually appealing, with a logo + application should provide important contact information of the IIC.

Every tab is designed separately for each screen. The tab on which the user is currently on is highlighted in a brighter red, so that the user is aware of where they are[16].

Every news item is organized and placed into its appropriate category. These sections are separated through tabs.

```
//initialize the quote screen - setup the layout
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.quote_screen);
    newsList = new ArrayList<HashMap<String, String>>();

    //buttons leading to the other activities in the application
    infoButton = (Button) findViewById(R.id.infoButton);
    infoButton.setOnClickListener((view) -> {
            Intent i = new Intent(view.getContext(), infoListScreen.class);
            i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
            overridePendingTransition(0,0); //0 for no animation
            startActivity(i);
    });
    eventButton = (Button) findViewById(R.id.eventButton);
    eventButton.setOnClickListener((view) -> {
            Intent i = new Intent(view.getContext(), eventListScreen.class);
            i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
            overridePendingTransition(0,0); //0 for no animation
            startActivity(i);
    });
    moreButton = (Button) findViewById(R.id.moreButton);
    moreButton.setOnClickListener((view) -> {
            Intent i = new Intent(view.getContext(), moreScreen.class);
            i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
            overridePendingTransition(0,0); //0 for no animation
            startActivity(i);
    });
```

```
<Button
    android:id="@+id/quoteButton"
    android:text="Quote"
    android:textColor="@color/white"
    android:textSize="15sp"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_weight="1"
    android:background="@color/clicked"/>

<Button
    android:id="@+id/infoButton"
    android:text="Information"
    android:textColor="@color/white"
    android:textSize="15sp"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_weight="1"
    android:background="@color/red"/>

<Button
    android:id="@+id/eventButton"
    android:text="Events"
    android:textColor="@color/white"
    android:textSize="15sp"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_weight="1"
    android:background="@color/red"/>

<Button
    android:id="@+id/moreButton"
    android:text="More"
    android:textColor="@color/white"
    android:textSize="15sp"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_weight="1"
    android:background="@color/red"/>

</LinearLayout>
```



A set of buttons to create the tabs, present under each screen. The button onClickListeners link to the different Java activates/screen once tapped. NOTE: buttons will change depending on the screen the user is already on

Tabs designed in XML. Same XML layout used for every screen. The screen the user is currently on will be a brighter red.

[16] Criterion B, Specification 2 and 3: Application must be user friendly and work smoothly with the user + application should be visually appealing, with a logo.

A custom ListView adapter was designed for news items to make the list elements more detailed and visually pleasing[17]. The new ListView displays the title, a description of the news, and states the type of item. This gives more information to the users and they do not have to keep clicking back and forth to see if the item they tapped has the information they need. The new custom design is efficient and helpful[18].

Default Android ListView elements were changed, and a custom XML based theme was applied

```
//Updating parsed JSON data into ListView using a custom adapter
//All listviews are made of data and adapters. The adapter handles the design, theme, and source of the information.
/*To create a custom adapter, a customized row.xml file was coded to make a theme of the individual listview elements,
and the data is provided using the JSON list.*/
ListAdapter adapter = new SimpleAdapter(
        eventListScreen.this, newsList,
        R.layout.row, new String[] { TITLE, SUBTITLE,
        NEWSTYPE,BODY }, new int[] { R.id.rowTitle,
        R.id.rowSubtitle, R.id.rowClass, R.id.rowBody });

setListAdapter(adapter);
```

All news items parsed through the JSON database file are displayed through this adapter.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="@drawable/row_select">

    <TextView
        android:id="@+id/rowTitle"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Title text here..."
        android:textColor="@color/rowTitle"
        android:textSize="21sp"
        android:paddingLeft="20dp"
        android:paddingStart="20dp"
        android:paddingRight="20dp"
        android:paddingEnd="20dp"
        android:paddingTop="6dp"
        android:paddingBottom="8dp"/>

    <TextView...>

    <TextView...>

    <TextView...>

    <ImageView...>
</RelativeLayout>
```

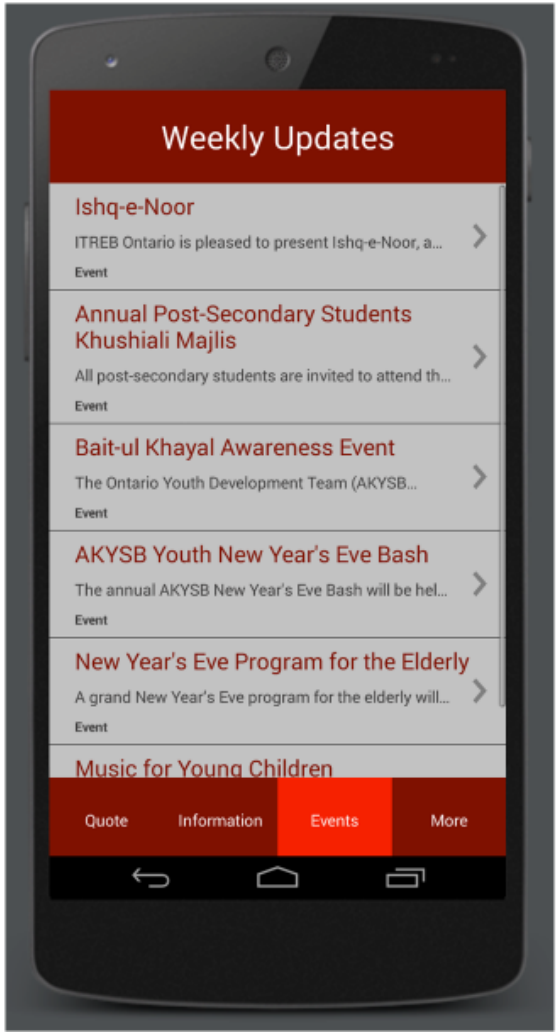
XML code used to design each individual element of one news item displayed in the list. The title, subtitle, and body of the news all have different themes.

[17] Android Custom ListView with Image and Text." Android Custom ListView with Image and Text. Accessed January 16, 2015.
[18] Criterion B, Specification 1 and 3: Create an application/system to display weekly news updates, event information, and Quote of the Week from Ismaili Institution Canada (IIC) + application should be visually appealing, with a logo.

Old Android ListView theme VS. the new custom XML based ListView adapter designed for the news list

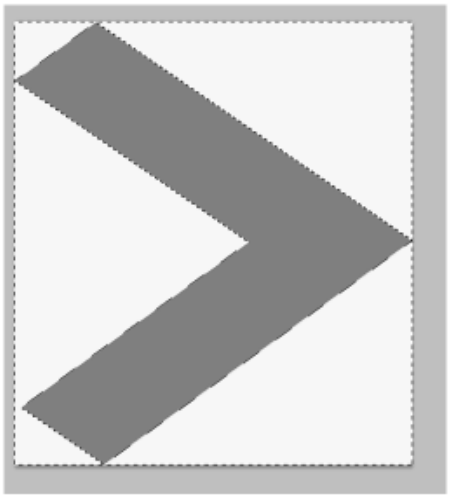
Old

New

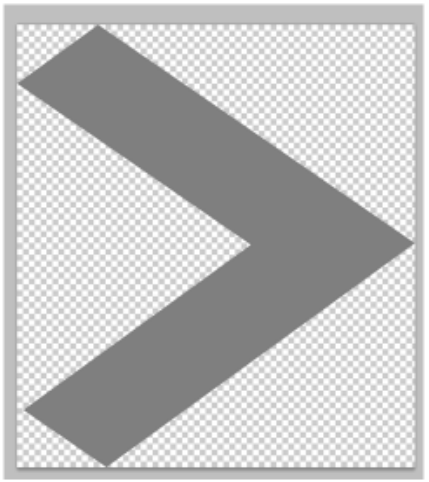


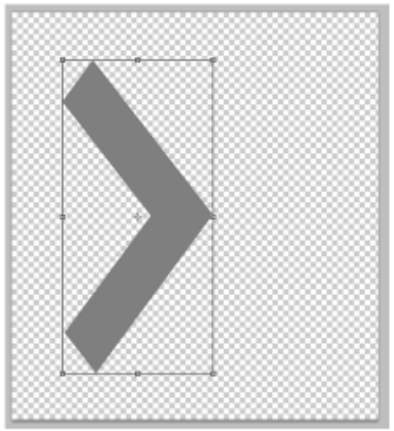
This graphic was manipulated in Photoshop[19]. The magic wand tool was used to remove the white background for transparency. It was scaled and skewed down so that the arrow has a slim design. This arrow was placed at the end of the custom ListView adapter to give it a “click to expand” type operation so that the user knows there are additional details.

Manipulation of graphics to aid the user. An arrow was added on the side of every news element.



Magic wand in Photoshop was used to delete the white background and make the arrow transparent





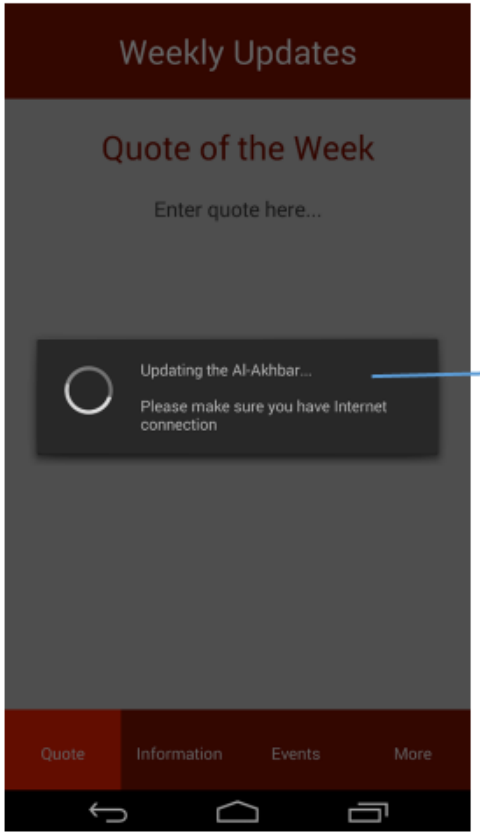
Arrow was scaled and resized to make it smaller. The slim design bring a clean design to every news item.

[19] "Right_grey_arrow Clip Art." Right Grey Arrow Clip Art. Accessed December 27, 2014. http://www.clker.com/clipart-right-grey-arrow.html

An important principle of design is to ensure the user is never left alone/confused. When the news items are being populated into the ListView, each device will take a different amount of processing time. A dialog box with a progress bar displayed while the ListView gets updated educates the user of all the processes – important for creating a fluid user experience[20].

Dialog boxes were placed while background thread downloadThread.java downloads the news

```
@Override
protected void onPreExecute() {
    super.onPreExecute();
    //launch a dialog box to keep main system process busy with GUI, while the background thread handles the file reading
    pDialog = new ProgressDialog(eventListScreen.this, ProgressDialog.THEME_HOLO_DARK);
    pDialog.setMessage("Updating the Al-Akhbar...\n\nPlease make sure you have Internet connection");
    pDialog.setCancelable(false);
    pDialog.show();
}
```



Progress bar helps keep the user updated on the download process of the news.

997 words

[20] Criterion B, Specification 2: Application must be user friendly and work smoothly with the user.

```java
package com.example.zain.alakhbar;

import android.content.Context;
import android.os.AsyncTask;
import android.util.Log;

import org.apache.http.util.ByteArrayBuffer;

import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.net.URL;
import java.net.URLConnection;

/**
 * Created by Zain on 12/25/2014.
 */

//download thread
public class downloadThread extends AsyncTask<Void, Void, String> {

    public downloadThread(Context context) {

    }

    @Override
    protected void onPreExecute() {

    }

    //perform these operations in the background thread
    @Override
    protected String doInBackground(Void... params) {

        String fileName = "thisweek.json";          //create this file on device
        String DownloadUrl = "http://zainzulfiqar.com/thisweek.json";       //download from this URL

        //download the file
        try {
            File root = android.os.Environment.getExternalStorageDirectory();

```

```
            //make a folder on the root directory, names AlAkhbar
            File dir = new File (root.getAbsolutePath() + "/AlAkhbar");
            if(dir.exists()==false) {
                dir.mkdirs();
            }

            URL url = new URL(DownloadUrl); //use the link specified before
            File file = new File(dir, fileName);

            //log what is happeneing for easy debugging
            long startTime = System.currentTimeMillis();
            Log.d("DownloadManager", "download begining");
            Log.d("DownloadManager", "download url:" + url);
            Log.d("DownloadManager", "downloaded file name:" + fileName);

           // Open a connection to that URL.
            URLConnection ucon = url.openConnection();

           //Define InputStreams to read from the URLConnection.
            InputStream is = ucon.getInputStream();
            BufferedInputStream bis = new BufferedInputStream(is);

           //Read bytes to the Buffer until there is nothing more to read.
            ByteArrayBuffer baf = new ByteArrayBuffer(5000);
            int current = 0;
            while ((current = bis.read()) != -1) {
                baf.append((byte) current);
            }

           // Convert the Bytes read to a String.
            FileOutputStream fos = new FileOutputStream(file);
            fos.write(baf.toByteArray());
            fos.flush();
            fos.close();
            Log.d("DownloadManager", "download ready in" + ((System.currentTimeMillis() - startTime) / 1000) + " sec");

        } catch (IOException e) {
            Log.d("DownloadManager", "Error: " + e);
        }

        return null;
    }

    @Override
    protected void onPostExecute(String result) {

    }
}
```

eventListScreen.java

```java
package com.example.zain.alakhbar;

import java.io.File;
import java.io.FileInputStream;
import java.nio.MappedByteBuffer;
import java.nio.channels.FileChannel;
import java.nio.charset.Charset;
import java.util.ArrayList;
import java.util.HashMap;

import org.json.JSONArray;
import org.json.JSONObject;

import android.app.ListActivity;
import android.app.ProgressDialog;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Environment;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.Button;
import android.widget.ListAdapter;
import android.widget.ListView;
import android.widget.SimpleAdapter;
import android.widget.TextView;

public class eventListScreen extends ListActivity {

    //private ProgressDialog pDialog;

    // URL to get JSON
    private static String url = "http://zainzulfiqar.com/thisweek.json";

    // JSON Node names
    private static final String TITLE = "title";
    private static final String BODY = "body";
    private static final String SUBTITLE = "subtitle";
    private static final String NEWSTYPE = "class";
    private static final String NEWSTHISWEEK = "newsThisWeek";

    private Button quoteButton = null;
    private Button infoButton = null;
    private Button moreButton = null;

    // news JSONArray
    JSONArray news = null;

    // Hashmap for ListView
    ArrayList<HashMap<String, String>> newsList;
    HashMap<String, String> newsItems = new HashMap<String, String>();

    //initialize the event screen - setup the layout
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.event_list);

        //buttons leading to the other activities in the application
        quoteButton = (Button) findViewById(R.id.quoteButton);
        quoteButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), quoteScreen.class);
                i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                overridePendingTransition(0,0); //0 for no animation
                startActivity(i);
            }
        });
        infoButton = (Button) findViewById(R.id.infoButton);
        infoButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), infoListScreen.class);
                i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                overridePendingTransition(0,0); //0 for no animation
                startActivity(i);
            }
        });
        moreButton = (Button) findViewById(R.id.moreButton);
        moreButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), moreScreen.class);
                i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                overridePendingTransition(0,0); //0 for no animation
                startActivity(i);
            }
        });

        //initialize the listview
```

```
        newsList = new ArrayList<HashMap<String, String>>();
        ListView lv = getListView();

        // Listview on item click listener, launch a seperate activity to show the expanded version of the news
        lv.setOnItemClickListener(new OnItemClickListener() {

            @Override
            public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
                // transfer the listview details to a new class to show individual expanded details

                String title = ((TextView) view.findViewById(R.id.rowTitle))
                        .getText().toString();
                String newsType = ((TextView) view.findViewById(R.id.rowClass))
                        .getText().toString();
                String body = ((TextView) view.findViewById(R.id.rowBody))
                        .getText().toString();

                Intent i = new Intent(view.getContext(), singleNews.class);
                String[] newsValues = new String[] {title, newsType, body};
                i.putExtra("newsData", newsValues);             //bundle it into a newsData named package - easily transfer information togeth
                startActivity(i);
            }
        });

        //launch the JSON processing script to populate the listview
        new GetNews().execute();
    }

    //Async task class to read JSON file from device
    private class GetNews extends AsyncTask<Void, Void, Void> {

        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            /*//launch a dialog box to keep main system process busy with GUI, while the background thread handles the file reading - also so
            pDialog = new ProgressDialog(eventListScreen.this, ProgressDialog.THEME_HOLO_DARK);
            pDialog.setMessage("Updating the Al-Akhbar...\n\nPlease make sure you have Internet connection");
            pDialog.setCancelable(false);
            pDialog.show();*/
        }

        //perform these operations in the background thread
        @Override
        protected Void doInBackground(Void... arg0) {
            // Creating the JSON string for data
            String jsonStr = null;

            Log.d("Response: ", "> " + jsonStr);

            //read the downloaded JSON file from online SQL server, and parse it into a string format
            try {
                File yourFile = new File(Environment.getExternalStorageDirectory(), "AlAkhbar/thisweek.json");
                FileInputStream stream = new FileInputStream(yourFile);
                try {
                    FileChannel fc = stream.getChannel();
                    MappedByteBuffer bb = fc.map(FileChannel.MapMode.READ_ONLY, 0, fc.size());

                    jsonStr = Charset.defaultCharset().decode(bb).toString();
                }
                finally {
                    stream.close();
                }

                //create a JSON object, to parse file
                JSONObject jsonObj = new JSONObject(jsonStr);

                // specify the table name (table name is an array, that contains all the element of the table) of the JSON file
                news = jsonObj.getJSONArray(NEWSTHISWEEK);

                // looping through all the elements in the table
                for (int i = 0; i < news.length(); i++) {
                    JSONObject c = news.getJSONObject(i);

                    String title = c.getString(TITLE);
                    String body = c.getString(BODY);
                    String subtitle = c.getString(SUBTITLE);
                    String newsType = c.getString(NEWSTYPE);

                    //use if statement to filter what content we want to display in this activity
                    if (newsType.equals("e")) {
                        newsType = "Event";

                        // tmp hashmap for single contact
                        newsItems = new HashMap<String, String>();

                        // adding each child node to HashMap key => value
                        newsItems.put(TITLE, title);
                        newsItems.put(BODY, body);
                        newsItems.put(SUBTITLE, subtitle);
                        newsItems.put(NEWSTYPE, newsType);

                        // adding news to news list
                        newsList.add(newsItems);
                    }
                }
            } catch (Exception e) {
                e.printStackTrace();
```

```
            }

            return null;
        }

        //after all of the above processes are done, launch the following
        @Override
        protected void onPostExecute(Void result) {
            super.onPostExecute(result);
            // Dismiss the progress dialog
            /*if (pDialog.isShowing())
                pDialog.dismiss();*/

            //Updating parsed JSON data into ListView using a custom adapter
            //All listviews are made of data and adapters. The adapter handles the design, theme, and source of the information.
            /*To create a custom adapter, a customized row.xml file was coded to make a theme of the individual listview elements,
            and the data is provided using the JSON list.*/
            ListAdapter adapter = new SimpleAdapter(
                    eventListScreen.this, newsList,
                    R.layout.row, new String[] { TITLE, SUBTITLE,
                    NEWSTYPE,BODY }, new int[] { R.id.rowTitle,
                    R.id.rowSubtitle, R.id.rowClass, R.id.rowBody });

            setListAdapter(adapter);
        }

    }
}
```

home.java

```java
package com.example.zain.alakhbar;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.Bundle;
import android.os.Environment;
import android.provider.Settings;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.os.Build;
import android.widget.Button;
import android.widget.ListView;
import android.widget.TextView;
import android.app.ActionBar;
import android.app.Fragment;
import android.view.LayoutInflater;
import android.view.ViewGroup;
import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.net.URL;
import java.net.URLConnection;
import org.apache.http.util.ByteArrayBuffer;
import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;


public class home extends Activity {

    //create a button object
    private Button menuButton = null;

    //initialize what happens at the start of this activity
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);         //use the XML activity_home layout file to setup the screen

        //setup the download thread - to update the news, "this" refers to launching it on the current activity/screen
        final downloadThread downloadNews = new downloadThread(this);

        //setup a listener for the button object and link it to the Enter button so that user can go to next screen
        menuButton = (Button) findViewById(R.id.menuButton);        //use the XML menuButton layout file to setup the button
        menuButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                File root = android.os.Environment.getExternalStorageDirectory();
                File file = new File (root.getAbsolutePath() + "/AlAkhbar/thisweek.json");         //check to see if file exists
                if(!file.exists() && !isConnected(home.this)) {         //if file is not there, and no Internet, this is a first time connection
                    System.out.println(""+file.exists());
                    buildDialog(home.this).show();
                }

```

```
                else if(file.exists() && !isConnected(home.this)) {                 //if file is there, and no Internet, you can still view the news stored on internal storage
                    Intent i = new Intent(view.getContext(), quoteScreen.class);
                    startActivity(i);
                }

                else if(isConnected(home.this)) {                   //if file is there, and connected to Internet, update to find latest file on database, and then view the news
                    downloadNews.execute();
                    Intent i = new Intent(view.getContext(), quoteScreen.class);    //setup the new activity, referring to the class file of the new screen
                    startActivity(i);       //start the new screen
                }
            }
        });
    }

    //check to see if connected to network
    public boolean isConnected(Context context) {

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

        if (netinfo != null && netinfo.isConnectedOrConnecting()) {
            android.net.NetworkInfo wifi = cm.getNetworkInfo(ConnectivityManager.TYPE_WIFI);
            android.net.NetworkInfo mobile = cm.getNetworkInfo(ConnectivityManager.TYPE_MOBILE);

            if((mobile != null && mobile.isConnectedOrConnecting()) || (wifi != null && wifi.isConnectedOrConnecting())) return true;
            else return false;
        } else
            return false;
    }

    //spawn an alert dialog to inform the user that there is no Internet connection
    public AlertDialog.Builder buildDialog(Context c) {

        AlertDialog.Builder builder = new AlertDialog.Builder(c, AlertDialog.THEME_HOLO_DARK);
        builder.setTitle("No Internet connection");
        builder.setMessage("Please connect to the internet to update the Al-Akhbar for first-time use.");

        builder.setPositiveButton("OK", new DialogInterface.OnClickListener() {

            @Override
            public void onClick(DialogInterface dialog, int which) {           //on clicking the OK button, do this...
                dialog.dismiss();
                startActivity(new Intent(Settings.ACTION_WIFI_SETTINGS));       //launch the Android system WiFi settings menu
            }
        });

        return builder;
    }
}

```

infoListScreen.java

```java
package com.example.zain.alakhbar;

import java.io.File;
import java.io.FileInputStream;
import java.nio.MappedByteBuffer;
import java.nio.channels.FileChannel;
import java.nio.charset.Charset;
import java.util.ArrayList;
import java.util.HashMap;

import org.json.JSONArray;
import org.json.JSONObject;

import android.app.ListActivity;
import android.app.ProgressDialog;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Environment;
import android.os.Parcelable;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.Button;
import android.widget.ListAdapter;
import android.widget.ListView;
import android.widget.SimpleAdapter;
import android.widget.TextView;

public class infoListScreen extends ListActivity {

    //private ProgressDialog pDialog;
    private static Parcelable mListViewScrollPos = null;

    // URL to get JSON
    private static String url = "http://zainzulfiqar.com/thisweek.json";

    // JSON Node names
    private static final String TITLE = "title";
    private static final String BODY = "body";
    private static final String SUBTITLE = "subtitle";
    private static final String NEWSTYPE = "class";
    private static final String NEWSTHISWEEK = "newsThisWeek";

    private Button quoteButton = null;
    private Button eventButton = null;
    private Button moreButton = null;

    // news JSONArray
    JSONArray news = null;

    // Hashmap for ListView
    ArrayList<HashMap<String, String>> newsList;
    HashMap<String, String> newsItems = new HashMap<String, String>();

    //initialize the info screen - setup the layout
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.info_list);

        //buttons leading to the other activities in the application
        quoteButton = (Button) findViewById(R.id.quoteButton);
        quoteButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), quoteScreen.class);
                i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                overridePendingTransition(0,0); //0 for no animation
                startActivity(i);
            }
        });
        eventButton = (Button) findViewById(R.id.eventButton);
        eventButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), eventListScreen.class);
                i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                overridePendingTransition(0,0); //0 for no animation
                startActivity(i);
            }
        });
        moreButton = (Button) findViewById(R.id.moreButton);
        moreButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), moreScreen.class);
                i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                overridePendingTransition(0,0); //0 for no animation
                startActivity(i);
            }
        });
```

```
94
95          //initialize the listview
96          newsList = new ArrayList<HashMap<String, String>>();
97          ListView lv = getListView();
98
99          // Listview on item click listener, launch a seperate activity to show the expanded version of the news
100         lv.setOnItemClickListener(new OnItemClickListener() {
101
102             @Override
103             public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
104                 // transfer the listview details to a new class to show individual expanded details
105
106                 String title = ((TextView) view.findViewById(R.id.rowTitle))
107                         .getText().toString();
108                 String newsType = ((TextView) view.findViewById(R.id.rowClass))
109                         .getText().toString();
110                 String body = ((TextView) view.findViewById(R.id.rowBody))
111                         .getText().toString();
112
113                 Intent i = new Intent(view.getContext(), singleNews.class);
114                 String[] newsValues = new String[] {title, newsType, body};
115                 i.putExtra("newsData", newsValues);             //bundle it into a newsData named package - easily transfer information togeth
116                 startActivity(i);
117             }
118         });
119
120         //launch the JSON processing script to populate the listview
121         new GetNews().execute();
122     }
123
124     //Async task class to read JSON file from device
125     private class GetNews extends AsyncTask<Void, Void, Void> {
126
127         @Override
128         protected void onPreExecute() {
129             super.onPreExecute();
130             /*//launch a dialog box to keep main system process busy with GUI, while the background thread handles the file reading - also so
131             pDialog = new ProgressDialog(infoListScreen.this, ProgressDialog.THEME_HOLO_DARK);
132             pDialog.setMessage("Updating the Al-Akhbar...\n\nPlease make sure you have Internet connection");
133             pDialog.setCancelable(false);
134             pDialog.show();*/
135         }
136
137         //perform these operations in the background thread
138         @Override
139         protected Void doInBackground(Void... arg0) {
140             // Creating the JSON string for data
141             String jsonStr = null;
142
143             Log.d("Response: ", "> " + jsonStr);
144
145             //read the downloaded JSON file from online SQL server, and parse it into a string format
146             try {
147                 File yourFile = new File(Environment.getExternalStorageDirectory(), "AlAkhbar/thisweek.json");
148                 FileInputStream stream = new FileInputStream(yourFile);
149                 try {
150                     FileChannel fc = stream.getChannel();
151                     MappedByteBuffer bb = fc.map(FileChannel.MapMode.READ_ONLY, 0, fc.size());
152
153                     jsonStr = Charset.defaultCharset().decode(bb).toString();
154                 }
155                 finally {
156                     stream.close();
157                 }
158
159                 //create a JSON object, to parse file
160                 JSONObject jsonObj = new JSONObject(jsonStr);
161
162                 // specify the table name (table name is an array, that contains all the element of the table) of the JSON file
163                 news = jsonObj.getJSONArray(NEWSTHISWEEK);
164
165                 // looping through all the elements in the table
166                 for (int i = 0; i < news.length(); i++) {
167                     JSONObject c = news.getJSONObject(i);
168
169                     //parse the appropriate information from the JSON file
170                         String title = c.getString(TITLE);
171                         String body = c.getString(BODY);
172                         String subtitle = c.getString(SUBTITLE);
173                         String newsType = c.getString(NEWSTYPE);
174
175                     //use if statement to filter what content we want to display in this activity
176                         if (newsType.equals("i")) {
177                             newsType = "Information";
178
179                             // tmp hashmap for single contact
180                             newsItems = new HashMap<String, String>();
181
182                             // adding each child node to HashMap key => value
183                             newsItems.put(TITLE, title);
184                             newsItems.put(BODY, body);
185                             newsItems.put(SUBTITLE, subtitle);
186                             newsItems.put(NEWSTYPE, newsType);
187
188                             // adding news to news list
189                             newsList.add(newsItems);
190                         }
```

```
                    }
                } catch (Exception e) {
                    e.printStackTrace();
                }

            return null;
        }

        //after all of the above processes are done, launch the following
        @Override
        protected void onPostExecute(Void result) {
            super.onPostExecute(result);
            // Dismiss the progress dialog
            /*if (pDialog.isShowing())
                pDialog.dismiss();*/

            //Updating parsed JSON data into ListView using a custom adapter
            //All listviews are made of data and adapters. The adapter handles the design, theme, and source of the information.
            /*To create a custom adapter, a customized row.xml file was coded to make a theme of the individual listview elements,
            and the data is provided using the JSON list.*/
            ListAdapter adapter = new SimpleAdapter(
                    infoListScreen.this, newsList,
                    R.layout.row, new String[] { TITLE, SUBTITLE,
                    NEWSTYPE,BODY }, new int[] { R.id.rowTitle,
                    R.id.rowSubtitle, R.id.rowClass, R.id.rowBody });

            setListAdapter(adapter);
        }
    }
}
```

moreScreen.java

```
1   package com.example.zain.alakhbar;
2
3   import android.app.Activity;
4   import android.content.Intent;
5   import android.os.AsyncTask;
6   import android.os.Bundle;
7   import android.os.Environment;
8   import android.text.Html;
9   import android.text.method.LinkMovementMethod;
10  import android.util.Log;
11  import android.view.View;
12  import android.widget.Button;
13  import android.widget.TextView;
14
15  import org.json.JSONArray;
16  import org.json.JSONObject;
17
18  import java.io.File;
19  import java.io.FileInputStream;
20  import java.nio.MappedByteBuffer;
21  import java.nio.channels.FileChannel;
22  import java.nio.charset.Charset;
23  import java.util.ArrayList;
24  import java.util.HashMap;
25
26  public class moreScreen extends Activity {
27
28      private Button infoButton = null;
29      private Button eventButton = null;
30      private Button quoteButton = null;
31
32      //initialize the quote screen - setup the layout
33      @Override
34      public void onCreate(Bundle savedInstanceState) {
35          super.onCreate(savedInstanceState);
36          setContentView(R.layout.more_screen);
37
38          //buttons leading to the other activities in the application
39          infoButton = (Button) findViewById(R.id.infoButton);
40          infoButton.setOnClickListener(new View.OnClickListener() {
41              @Override
```

```
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), infoListScreen.class);
                i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                overridePendingTransition(0,0); //0 for no animation
                startActivity(i);
            }
        });
        eventButton = (Button) findViewById(R.id.eventButton);
        eventButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), eventListScreen.class);
                i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                overridePendingTransition(0,0); //0 for no animation
                startActivity(i);
            }
        });
        quoteButton = (Button) findViewById(R.id.quoteButton);
        quoteButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), quoteScreen.class);
                i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                overridePendingTransition(0,0); //0 for no animation
                startActivity(i);
            }
        });

        TextView body = (TextView) findViewById(R.id.moreBody);
        body.setText(Html.fromHtml("<b>Ismaili Council for Canada (IIC)</b><br>" +
                "49 Wynford Drive<br>" +
                "Toronto, Ontario M3C 1K1<br>" +
                "CANADA<br>" +
                "Tel: +1 (416) 646-6965<br><br>" +
                "<b>Ismaili Council for Ontario</b><br>" +
                "149-151 Bartley Avenue<br>" +
                "Toronto, Ontario M4A 1C9<br>" +
                "CANADA<br>" +
                "Tel: +1-416-751-4001<br><br><br><br>" +
                "<i>Application developed by Zain Zulfiqar</i>"));
    }
}
```

quoteScreen.java

```java
package com.example.zain.alakhbar;

import android.app.Activity;
import android.app.ListActivity;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Environment;
import android.os.Handler;
import android.text.Html;
import android.text.method.LinkMovementMethod;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.Button;
import android.widget.ListAdapter;
import android.widget.ListView;
import android.widget.SimpleAdapter;
import android.widget.TextView;
import android.widget.Toast;

import org.json.JSONArray;
import org.json.JSONObject;

import java.io.File;
import java.io.FileInputStream;
import java.nio.MappedByteBuffer;
import java.nio.channels.FileChannel;
import java.nio.charset.Charset;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Set;

public class quoteScreen extends Activity {

    //private ProgressDialog pDialog;

    // URL to get JSON
    private static String url = "http://zainzulfiqar.com/thisweek.json";

    // JSON Node names
    private static final String BODY = "body";
    private static final String NEWSTYPE = "class";
    private static final String NEWSTHISWEEK = "newsThisWeek";

    private Button infoButton = null;
    private Button eventButton = null;
    private Button moreButton = null;

    // news JSONArray
    JSONArray news = null;

    // Hashmap for ListView
    ArrayList<HashMap<String, String>> newsList;
    HashMap<String, String> newsItems = new HashMap<String, String>();

    //initialize the quote screen - setup the layout
    @Override
    public void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.quote_screen);
        newsList = new ArrayList<HashMap<String, String>>();

        //buttons leading to the other activities in the application
        infoButton = (Button) findViewById(R.id.infoButton);
        infoButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), infoListScreen.class);
                i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                overridePendingTransition(0,0); //0 for no animation
                startActivity(i);
            }
        });
        eventButton = (Button) findViewById(R.id.eventButton);
        eventButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), eventListScreen.class);
                i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                overridePendingTransition(0,0); //0 for no animation
                startActivity(i);
            }
        });
        moreButton = (Button) findViewById(R.id.moreButton);
        moreButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), moreScreen.class);
                i.addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
                overridePendingTransition(0,0); //0 for no animation
                startActivity(i);
            }
        });

        //launch the JSON processing script to populate the listview
        new GetNews().execute();
    }

    //when back button pressed on this activity
    @Override
    public void onBackPressed() {
        //leave blank so that you can't go back to the home screen (going back causes a task execution error)
    }

    //Async task class to read JSON file from device
    private class GetNews extends AsyncTask<Void, Void, Void> {

        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            /*//launch a dialog box to keep main system process busy with GUI, while the background thread handles the file reading - also so that user knows what is going on
            pDialog = new ProgressDialog(quoteScreen.this, ProgressDialog.THEME_HOLO_DARK);
            pDialog.setMessage("Updating the Al-Akhbar...\n\nPlease make sure you have Internet connection");
            pDialog.setCancelable(false);
            pDialog.show();*/
        }

        //perform these operations in the background thread
        @Override
        protected Void doInBackground(Void... arg0) {
            // Creating the JSON string for data
            String jsonStr = null;

            //display log responses so it is easy to debug application
            Log.d("Response: ", "> " + jsonStr);

```

```
            //read the downloaded JSON file from online SQL server, and parse it into a string format
            try {
                File yourFile = new File(Environment.getExternalStorageDirectory(), "/AlAkhbar/thisweek.json");
                FileInputStream stream = new FileInputStream(yourFile);
                try {
                    FileChannel fc = stream.getChannel();
                    MappedByteBuffer bb = fc.map(FileChannel.MapMode.READ_ONLY, 0, fc.size());

                    jsonStr = Charset.defaultCharset().decode(bb).toString();
                }
                finally {
                    stream.close();
                }

                //create a JSON object, to parse file
                JSONObject jsonObj = new JSONObject(jsonStr);

                // specify the table name (table name is an array, that contains all the element of the table) of the JSON file
                news = jsonObj.getJSONArray(NEWSTHISWEEK);

                // looping through all the elements in the table
                for (int i = 0; i < news.length(); i++) {
                    JSONObject c = news.getJSONObject(i);

                    //parse the appropriate information from the JSON file
                    String body = c.getString(BODY);
                    String newsType = c.getString(NEWSTYPE);

                    //use if statement to filter what content we want to display in this activity
                    if (newsType.equals("q")) {
                        // tmp hashmap for single contact
                        newsItems = new HashMap<String, String>();

                        // adding each child node to HashMap key => value
                        newsItems.put(BODY, body);

                        // adding news to news list
                        newsList.add(newsItems);
                    }
                }
            } catch (Exception e) {
                e.printStackTrace();
            }

            return null;
        }

        //after all of the above processes are done, launch the following
        @Override
        protected void onPostExecute(Void result) {
            super.onPostExecute(result);
            // Dismiss the progress dialog
            /*if (pDialog.isShowing())
                pDialog.dismiss();*/

            //Updating parsed JSON data into ListView
            String quoteBodyValue = newsItems.get(BODY);
            TextView body = (TextView) findViewById(R.id.quoteBody);

            /*set an HTML element on Java string so that HTML tags can be recognized
            this allows for the data to be uniquely customized/styled by typing in HTML tags in the database itself*/
            body.setText(Html.fromHtml(quoteBodyValue));
            body.setMovementMethod(LinkMovementMethod.getInstance());
        }
    }
}
```

singleNews.java

```java
package com.example.zain.alakhbar;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.text.Html;
import android.text.method.LinkMovementMethod;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.os.Build;
import android.widget.Button;
import android.widget.ListView;
import android.widget.TextView;
import android.app.ActionBar;
import android.app.Fragment;
import android.view.LayoutInflater;
import android.view.ViewGroup;
import android.widget.Toast;

import java.util.ArrayList;


public class singleNews extends Activity {

    private Button backButton = null;

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

        backButton = (Button) findViewById(R.id.backButton);

        backButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                finish();       //use finish to end this activity, and go back to the previous one. This continues off the state of the previo
            }
        });

        //get data
        Intent i = getIntent();
        String[] newsExpandedValues = i.getStringArrayExtra("newsData");

        //update
        TextView title = (TextView) findViewById(R.id.singleTitle);
        TextView newsType = (TextView) findViewById(R.id.singleClass);
        TextView body = (TextView) findViewById(R.id.singleBody);
        title.setText(newsExpandedValues[0]);
        newsType.setText(newsExpandedValues[1]);

        // body setup
        body.setText(Html.fromHtml(newsExpandedValues[2]));
        body.setMovementMethod(LinkMovementMethod.getInstance());
    }
}

```

Screen snapshots of the XML code

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.zain.alakhbar" >
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.READ_PHONE_STATE"/>
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@android:style/Theme.NoTitleBar.Fullscreen" >
        <activity
            android:name=".home"
            android:label="@string/app_name"
            android:screenOrientation="portrait">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".quoteScreen"
            android:label="@string/title_activity_list_of_news"
            android:screenOrientation="portrait">
        </activity>
        <activity
            android:name=".infoListScreen"
            android:label="@string/title_activity_list_of_news"
            android:screenOrientation="portrait">
        </activity>
        <activity
            android:name=".eventListScreen"
            android:label="@string/title_activity_list_of_news"
            android:screenOrientation="portrait">
        </activity>
        <activity
            android:name=".moreScreen"
            android:label="@string/title_activity_list_of_news"
            android:screenOrientation="portrait">
        </activity>
        <activity
            android:name=".singleNews"
            android:label="@string/title_activity_list_of_news"
            android:screenOrientation="portrait">
        </activity>
    </application>
</manifest>

```

activity_home.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:background="#9C1B07"
    tools:context=".home">

    <TextView
        android:text="@string/menu_title1"
        android:textColor="#FFFFFF"
        android:textSize="35sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:layout_above="@+id/title2"/>

    <TextView
        android:id="@+id/title2"
        android:text="@string/menu_title2"
        android:textColor="#FFFFFF"
        android:textSize="66sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:weightSum="2"
        android:orientation="vertical">

        <TextView
            android:text=" "
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_weight="1"/>

        <LinearLayout
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:layout_weight="1"
            android:gravity="center_horizontal|center_vertical">

            <Button
            android:id="@+id/menuButton"
            android:layout_gravity="center"
            android:layout_width="wrap_content"
            android:layout_height="50dp"
            android:text="    Enter    "
            android:textSize="22sp"
```

```
57              android:gravity="center_horizontal|center_vertical"
58              android:textColor="#FFFFFF"
59              android:background="@drawable/home_button_define"/>
60
61
62          </LinearLayout>
63
64      </LinearLayout>
65
66      <TextView
67          android:id="@+id/testMode"
68          android:text="Version 1.00"
69          android:textColor="#FFFFFF"
70          android:textSize="17sp"
71          android:layout_width="wrap_content"
72          android:layout_height="wrap_content"
73          android:layout_centerHorizontal="true"
74          android:layout_centerVertical="true"
75          android:layout_alignParentBottom="true"
76          android:textStyle="italic"
77          android:layout_marginBottom="20dp"/>
78
79      <TextView
80          android:text="Z.Z."
81          android:textColor="#FFFFFF"
82          android:textSize="10sp"
83          android:layout_width="wrap_content"
84          android:layout_height="wrap_content"
85          android:layout_centerHorizontal="true"
86          android:layout_centerVertical="true"
87          android:layout_alignParentBottom="true"
88          android:textStyle="italic"/>
89
90  </RelativeLayout>
```

activity_news_expanded.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content">

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:weightSum="20">

        <Button
            android:id="@+id/backButton"
            android:layout_weight="17"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:text=" &#60; "
            android:textColor="@color/white"
            android:textSize="30sp"
            android:background="@drawable/home_button_define"/>

        <TextView android:id="@+id/headerNews"
            android:layout_weight="3"
            android:text="Back to News Feed"
            android:layout_height="wrap_content"
            android:layout_width="fill_parent"
            android:textColor="@color/white"
            android:background="@color/rowTitle"
            android:gravity="left"
            android:textSize="20sp"
            android:paddingTop="20dp"
            android:paddingBottom="20dp"/>
    </LinearLayout>

    <ScrollView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:background="@color/rowback">
        <LinearLayout
            android:orientation="vertical" android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:id="@+id/singleTitle"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Title Here..."
                android:textSize="25dp"
                android:textColor="@color/rowTitle"
                android:gravity="center"
                android:layout_marginEnd="10dp"
                android:layout_marginStart="10dp"
                android:paddingTop="15dp"
                android:layout_marginBottom="7dp"/>
```

```
57
58              <TextView
59                  android:id="@+id/singleClass"
60                  android:layout_width="match_parent"
61                  android:layout_height="wrap_content"
62                  android:text="Class Here..."
63                  android:textSize="15dp"
64                  android:textStyle="bold"
65                  android:textColor="@color/rowSubtitle"
66                  android:gravity="center"
67                  android:layout_marginBottom="10dp"/>
68
69              <View
70                  android:id="@+id/colored_bar"
71                  android:layout_width="fill_parent"
72                  android:layout_height="3dp"
73                  android:background="@color/red"
74                  android:layout_marginLeft="17dp"
75                  android:layout_marginRight="17dp"/>
76
77              <TextView
78                  android:id="@+id/singleBody"
79                  android:layout_width="match_parent"
80                  android:layout_height="wrap_content"
81                  android:text="Body Here..."
82                  android:textSize="15dp"
83                  android:textColor="@color/rowSubtitle"
84                  android:layout_marginTop="15dp"
85                  android:layout_marginBottom="20dp"
86                  android:layout_marginLeft="15dp"
87                  android:layout_marginRight="15dp"
88                  android:textColorHighlight="@color/rowTitle"
89                  android:textColorHint="@color/white"/>
90
91          </LinearLayout>
92      </ScrollView>
93  </LinearLayout>
```

event_list.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="wrap_content"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:id="@+id/rlLayout"
    android:background="@color/rowback"
    android:weightSum="20">

    <TextView android:id="@+id/headerNews"
        android:text="Weekly Updates"
        android:layout_height="wrap_content"
        android:layout_width="fill_parent"
        android:textColor="@color/white"
        android:background="@color/rowTitle"
        android:gravity="center"
        android:textSize="30sp"
        android:padding="20dp"
        android:layout_weight="0"/>

    <ListView
        android:id="@android:id/list"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_below="@+id/headerNews"
        android:smoothScrollbar="true"
        android:choiceMode="singleChoice"
        android:layout_weight="4">
    </ListView>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@android:id/list"
        android:layout_marginBottom="-6dp"
        android:orientation="horizontal"
        android:layout_weight="16"
        android:weightSum="4">

        <Button
            android:id="@+id/quoteButton"
            android:text="Quote"
            android:textColor="@color/white"
            android:textSize="15sp"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:layout_weight="1"
            android:background="@color/red"/>

        <Button
            android:id="@+id/infoButton"
            android:text="Information"
            android:textColor="@color/white"
            android:textSize="15sp"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
```

```
57                android:layout_weight="1"
58                android:background="@color/red"/>
59
60            <Button
61                android:id="@+id/eventButton"
62                android:text="Events"
63                android:textColor="@color/white"
64                android:textSize="15sp"
65                android:layout_width="fill_parent"
66                android:layout_height="fill_parent"
67                android:layout_weight="1"
68                android:background="@color/clicked"/>
69
70            <Button
71                android:id="@+id/moreButton"
72                android:text="More"
73                android:textColor="@color/white"
74                android:textSize="15sp"
75                android:layout_width="fill_parent"
76                android:layout_height="fill_parent"
77                android:layout_weight="1"
78                android:background="@color/red"/>
79
80            </LinearLayout>
81
82    </LinearLayout>
```

info_list.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="wrap_content"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:id="@+id/rlLayout"
    android:background="@color/rowback"
    android:weightSum="20">

    <TextView android:id="@+id/headerNews"
        android:text="Weekly Updates"
        android:layout_height="wrap_content"
        android:layout_width="fill_parent"
        android:textColor="@color/white"
        android:background="@color/rowTitle"
        android:gravity="center"
        android:textSize="30sp"
        android:padding="20dp"
        android:layout_weight="0"/>

    <ListView
        android:id="@android:id/list"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_below="@+id/headerNews"
        android:smoothScrollbar="true"
        android:choiceMode="singleChoice"
        android:layout_weight="4">
    </ListView>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@android:id/list"
        android:layout_marginBottom="-6dp"
        android:orientation="horizontal"
        android:layout_weight="16"
        android:weightSum="4">

        <Button
            android:id="@+id/quoteButton"
            android:text="Quote"
            android:textColor="@color/white"
            android:textSize="15sp"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:layout_weight="1"
            android:background="@color/red"/>

        <Button
            android:id="@+id/infoButton"
            android:text="Information"
            android:textColor="@color/white"
            android:textSize="15sp"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
```

```
57                android:layout_weight="1"
58                android:background="@color/clicked"/>
59
60            <Button
61                android:id="@+id/eventButton"
62                android:text="Events"
63                android:textColor="@color/white"
64                android:textSize="15sp"
65                android:layout_width="fill_parent"
66                android:layout_height="fill_parent"
67                android:layout_weight="1"
68                android:background="@color/red"/>
69
70            <Button
71                android:id="@+id/moreButton"
72                android:text="More"
73                android:textColor="@color/white"
74                android:textSize="15sp"
75                android:layout_width="fill_parent"
76                android:layout_height="fill_parent"
77                android:layout_weight="1"
78                android:background="@color/red"/>
79
80            </LinearLayout>
81
82    </LinearLayout>
```

more_screen.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:background="@color/rowback"
    android:weightSum="20">

    <TextView android:id="@+id/headerNews"
        android:text="Weekly Updates"
        android:layout_height="wrap_content"
        android:layout_width="fill_parent"
        android:textColor="@color/white"
        android:background="@color/rowTitle"
        android:gravity="center"
        android:textSize="30sp"
        android:padding="20dp"
        android:layout_weight="0"/>

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:orientation="vertical"
        android:layout_weight="4">

        <TextView
            android:id="@+id/moreTitle"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="More Details"
            android:gravity="center"
            android:textColor="@color/rowTitle"
            android:textSize="30sp"
            android:paddingTop="20dp"
            android:paddingBottom="20dp"/>

        <TextView
            android:id="@+id/moreBody"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="details here..."
            android:textColor="@color/rowSubtitle"
            android:textSize="19sp"
            android:gravity="center"
            android:layout_marginBottom="20dp"
            android:layout_marginLeft="20dp"
            android:layout_marginRight="20dp"/>

        </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@android:id/list"
        android:layout_marginBottom="-6dp"
        android:orientation="horizontal"
```

```
57            android:layout_weight="16"
58            android:weightSum="4">
59
60            <Button
61                android:id="@+id/quoteButton"
62                android:text="Quote"
63                android:textColor="@color/white"
64                android:textSize="15sp"
65                android:layout_width="fill_parent"
66                android:layout_height="fill_parent"
67                android:layout_weight="1"
68                android:background="@color/red"/>
69
70            <Button
71                android:id="@+id/infoButton"
72                android:text="Information"
73                android:textColor="@color/white"
74                android:textSize="15sp"
75                android:layout_width="fill_parent"
76                android:layout_height="fill_parent"
77                android:layout_weight="1"
78                android:background="@color/red"/>
79
80            <Button
81                android:id="@+id/eventButton"
82                android:text="Events"
83                android:textColor="@color/white"
84                android:textSize="15sp"
85                android:layout_width="fill_parent"
86                android:layout_height="fill_parent"
87                android:layout_weight="1"
88                android:background="@color/red"/>
89
90            <Button
91                android:id="@+id/moreButton"
92                android:text="More"
93                android:textColor="@color/white"
94                android:textSize="15sp"
95                android:layout_width="fill_parent"
96                android:layout_height="fill_parent"
97                android:layout_weight="1"
98                android:background="@color/clicked"/>
99
100           </LinearLayout>
101
102   </LinearLayout>
```

quote_screen.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:background="@color/rowback"
    android:weightSum="20">

    <TextView android:id="@+id/headerNews"
        android:text="Weekly Updates"
        android:layout_height="wrap_content"
        android:layout_width="fill_parent"
        android:textColor="@color/white"
        android:background="@color/rowTitle"
        android:gravity="center"
        android:textSize="30sp"
        android:padding="20dp"
        android:layout_weight="0"/>

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:orientation="vertical"
        android:layout_weight="4">

        <TextView
            android:id="@+id/quoteTitle"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="Quote of the Week"
            android:gravity="center"
            android:textColor="@color/rowTitle"
            android:textSize="30sp"
            android:paddingTop="20dp"
            android:paddingBottom="20dp"/>

        <TextView
            android:id="@+id/quoteBody"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="Loading quote..."
            android:textColor="@color/rowSubtitle"
            android:textSize="19sp"
            android:gravity="center"
            android:layout_marginBottom="20dp"
            android:layout_marginLeft="20dp"
            android:layout_marginRight="20dp"/>

        </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@android:id/list"
        android:layout_marginBottom="-6dp"
        android:orientation="horizontal"
```

```
57            android:layout_weight="16"
58            android:weightSum="4">
59
60            <Button
61                android:id="@+id/quoteButton"
62                android:text="Quote"
63                android:textColor="@color/white"
64                android:textSize="15sp"
65                android:layout_width="fill_parent"
66                android:layout_height="fill_parent"
67                android:layout_weight="1"
68                android:background="@color/clicked"/>
69
70            <Button
71                android:id="@+id/infoButton"
72                android:text="Information"
73                android:textColor="@color/white"
74                android:textSize="15sp"
75                android:layout_width="fill_parent"
76                android:layout_height="fill_parent"
77                android:layout_weight="1"
78                android:background="@color/red"/>
79
80            <Button
81                android:id="@+id/eventButton"
82                android:text="Events"
83                android:textColor="@color/white"
84                android:textSize="15sp"
85                android:layout_width="fill_parent"
86                android:layout_height="fill_parent"
87                android:layout_weight="1"
88                android:background="@color/red"/>
89
90            <Button
91                android:id="@+id/moreButton"
92                android:text="More"
93                android:textColor="@color/white"
94                android:textSize="15sp"
95                android:layout_width="fill_parent"
96                android:layout_height="fill_parent"
97                android:layout_weight="1"
98                android:background="@color/red"/>
99
100           </LinearLayout>
101
102   </LinearLayout>
```

row.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="@drawable/row_select">

    <TextView
        android:id="@+id/rowTitle"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Title text here..."
        android:textColor="@color/rowTitle"
        android:textSize="21sp"
        android:paddingLeft="20dp"
        android:paddingStart="20dp"
        android:paddingRight="20dp"
        android:paddingEnd="20dp"
        android:paddingTop="6dp"
        android:paddingBottom="8dp"/>

    <TextView
        android:id="@+id/rowSubtitle"
        android:layout_below="@+id/rowTitle"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Subitle text here..."
        android:textColor="@color/rowSubtitle"
        android:textSize="15sp"
        android:paddingLeft="20dp"
        android:paddingStart="20dp"
        android:paddingBottom="9dp"
        android:paddingRight="40dp"
        android:paddingEnd="40dp"
        android:maxLines="1"
        android:ellipsize="end"/>

    <TextView
        android:id="@+id/rowClass"
        android:layout_below="@+id/rowSubtitle"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Quote, event, info, ad?"
        android:textColor="@color/rowSubtitle"
        android:textSize="12sp"
        android:textStyle="bold"
        android:paddingLeft="20dp"
        android:paddingStart="20dp"
        android:paddingBottom="7dp"
        android:paddingRight="40dp"
        android:paddingEnd="40dp"/>

    <TextView
        android:id="@+id/rowBody"
        android:layout_below="@+id/rowClass"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
57            android:text="The body here is hidden..."
58            android:visibility="gone"
59            android:textColor="@color/rowSubtitle"
60            android:textSize="12sp"
61            android:textStyle="bold"
62            android:paddingLeft="20dp"
63            android:paddingStart="20dp"
64            android:paddingBottom="15dp"
65            android:paddingRight="40dp"
66            android:paddingEnd="40dp"/>
67
68        <ImageView
69            android:padding="15dp"
70            android:layout_width="41dp"
71            android:layout_height="50dp"
72            android:adjustViewBounds="true"
73            android:scaleType="fitXY"
74            android:src="@drawable/right_arrow"
75            android:layout_alignParentEnd="true"
76            android:layout_alignParentRight="true"
77            android:layout_centerInParent="true"
78            />
79    </RelativeLayout>
```