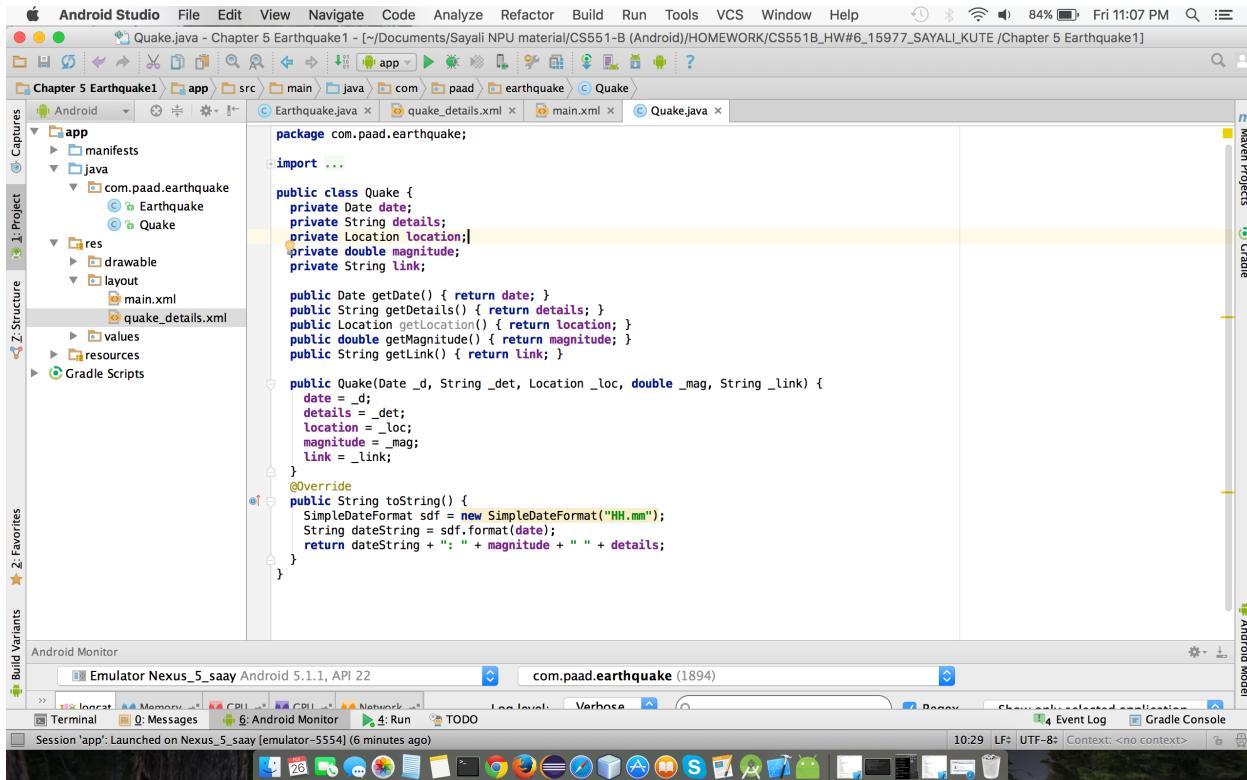


Name: Sayali Vishnu Kute

Email ID: [sayalikute2507@gmail.com](mailto:sayalikute2507@gmail.com)

## Implementation of Earthquake Viewer App on Android Studio

Create Earthquake.java class as follows:



```
package com.paad.earthquake;

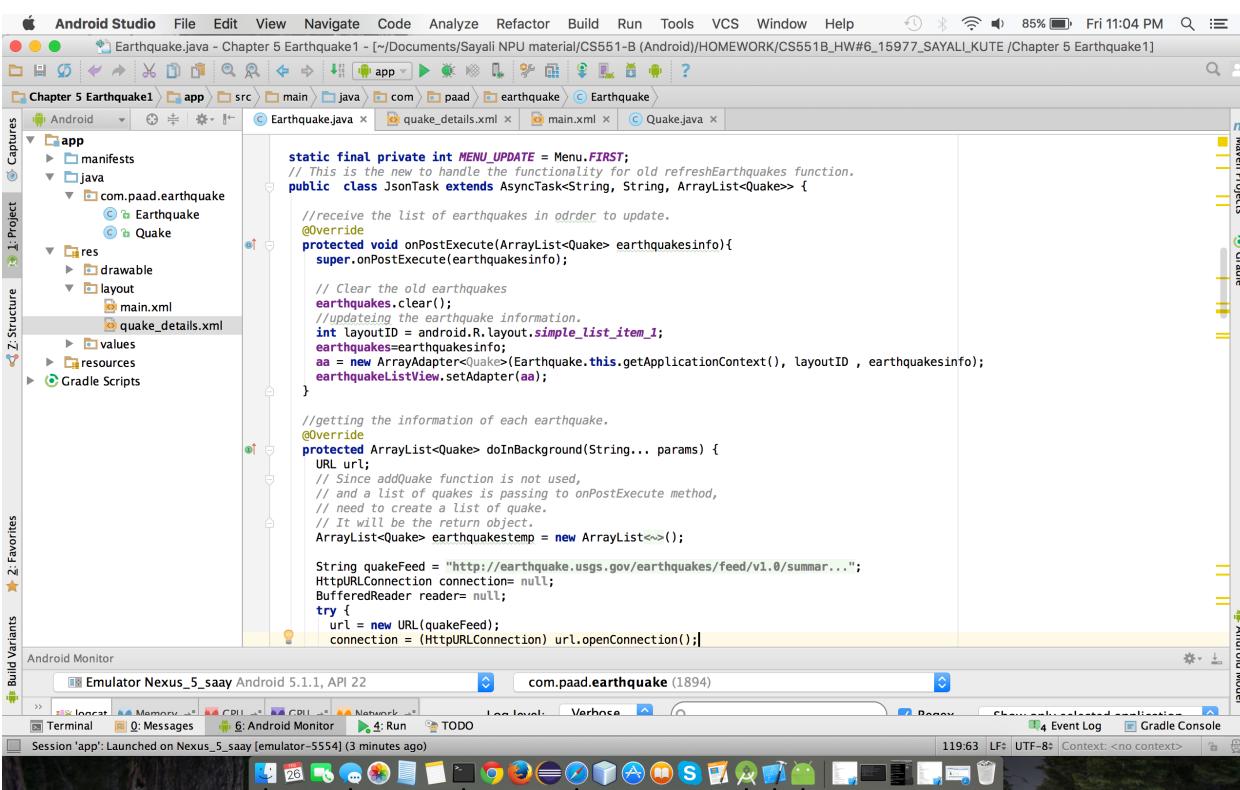
import ...

public class Quake {
    private Date date;
    private String details;
    private Location location;
    private double magnitude;
    private String link;

    public Date getDate() { return date; }
    public String getDetails() { return details; }
    public Location getLocation() { return location; }
    public double getMagnitude() { return magnitude; }
    public String getLink() { return link; }

    public Quake(Date _date, String _det, Location _loc, double _mag, String _link) {
        date = _date;
        details = _det;
        location = _loc;
        magnitude = _mag;
        link = _link;
    }

    @Override
    public String toString() {
        SimpleDateFormat sdf = new SimpleDateFormat("HH:mm");
        String dateString = sdf.format(date);
        return dateString + ":" + magnitude + " " + details;
    }
}
```



```
static final private int MENU_UPDATE = Menu.FIRST;
// This is the new handle the functionality for old refreshEarthquakes function.
public class JsonTask extends AsyncTask<String, String, ArrayList<Quake>> {

    //receive the list of earthquakes in order to update.
    @Override
    protected void onPostExecute(ArrayList<Quake> earthquakesinfo){
        super.onPostExecute(earthquakesinfo);

        // Clear the old earthquakes
        earthquakes.clear();
        //updating the earthquake information.
        int layoutID = android.R.layout.simple_list_item_1;
        earthquakes=earthquakesinfo;
        aa = new ArrayAdapter<Quake>(Earthquake.this.getApplicationContext(), layoutID , earthquakesinfo);
        earthquakeListView.setAdapter(aa);
    }

    //getting the information of each earthquake.
    @Override
    protected ArrayList<Quake> doInBackground(String... params) {
        URL url;
        // Since addQuake function is not used,
        // and a list of quakes is passing to onPostExecute method,
        // need to create a list of quake.
        // It will be the return object.
        ArrayList<Quake> earthquakestemp = new ArrayList<Quake>();

        String quakeFeed = "http://earthquake.usgs.gov/earthquakes/feed/v1.0/summary...";
        HttpURLConnection connection= null;
        BufferedReader reader= null;
        try {
            url = new URL(quakeFeed);
            connection = (HttpURLConnection) url.openConnection();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Name: Sayali Vishnu Kute

Email ID: [sayalikute2507@gmail.com](mailto:sayalikute2507@gmail.com)

The screenshot shows the Android Studio interface with the Earthquake.java file open. The code reads a JSON feed from a URL and extracts specific information like date, details, and location.

```
url = new URL(quakeFeed);
connection = (HttpURLConnection) url.openConnection();
connection.connect();

//prepare to read the json
InputStream stream = connection.getInputStream();
reader = new BufferedReader(new InputStreamReader(stream));
StringBuffer buffer = new StringBuffer();

String line = "";
// start to read the earthquakes!!!
while((line=reader.readLine())!=null){

    // To chop off the unnecessary information because the url is jsonp not json and there has a header.
    // each line will contain one entry of earthquake information.
    line=line.substring(line.indexOf("{\"type\":\"Feature\""),line.lastIndexOf("}")+1);
    /* Testing entry to make sure Quake works properly.
    */
    String details = "16km S of Medford, Oklahoma";
    details = details.split(",")[1].trim();
    String hostname = "http://earthquake.usgs.gov";
    String linkString = hostname + " http://earthquake.usgs.gov/earthquakes/eventpage/us10004s75";
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd'T'hh:mm:ss'Z");

    Date qdate = new GregorianCalendar(0,0,0).getTime();
    try {
        qdate = sdf.parse("2010-10-19T21:39:35Z");
    } catch (ParseException e) {
        e.printStackTrace();
    }
    long time = Long.parseLong("1456269504030");

    /*
    */

    // getting the jsonobject
    JSONObject object = new JSONObject(line);
    // getting elements which contain the wanted information.
    JSONObject properties= object.getJSONObject("properties");
    JSONObject geometry = object.getJSONObject("geometry");

    // getting the update time.
    // in json it is stored in long int format and need to parse it to date.
    Date qdate = new Date(Long.parseLong(properties.getString("updated")));

    // getting the details.
    // to follow the original idea modify it to state's name only.
    String details = properties.getString("place");
    details = details.substring(details.indexOf(",")+2);

    // getting the location
    Location l = new Location("dummyGPS");
    JSONArray coordinate = geometry.getJSONArray("coordinates");
    l.setLatitude(Double.parseDouble(coordinate.getString(0)));
    l.setLongitude(Double.parseDouble(coordinate.getString(1)));

    // getting the url
    // to follow the original design only change the second part of linkString
    // modify a little for the host to make the dialog looks prettier.
    String hostname = "http://earthquake.usgs.gov ";
    String linkString = hostname + properties.getString("url");

    // getting the magnitude
    double magnitude = Double.parseDouble(properties.getString("mag"));

    // create a new entry of quake
    Quake quake = new Quake(qdate, details, l, magnitude, linkString);
    // Process a newly found earthquake
}
```

The screenshot shows the modified Earthquake.java code where the JSON data is being processed to create a new Quake object and handle it.

```
/*
 */

// getting the jsonobject
JSONObject object = new JSONObject(line);
// getting elements which contain the wanted information.
JSONObject properties= object.getJSONObject("properties");
JSONObject geometry = object.getJSONObject("geometry");

// getting the update time.
// in json it is stored in long int format and need to parse it to date.
Date qdate = new Date(Long.parseLong(properties.getString("updated")));

// getting the details.
// to follow the original idea modify it to state's name only.
String details = properties.getString("place");
details = details.substring(details.indexOf(",")+2);

// getting the location
Location l = new Location("dummyGPS");
JSONArray coordinate = geometry.getJSONArray("coordinates");
l.setLatitude(Double.parseDouble(coordinate.getString(0)));
l.setLongitude(Double.parseDouble(coordinate.getString(1)));

// getting the url
// to follow the original design only change the second part of linkString
// modify a little for the host to make the dialog looks prettier.
String hostname = "http://earthquake.usgs.gov ";
String linkString = hostname + properties.getString("url");

// getting the magnitude
double magnitude = Double.parseDouble(properties.getString("mag"));

// create a new entry of quake
Quake quake = new Quake(qdate, details, l, magnitude, linkString);
// Process a newly found earthquake
}
```

Name: Sayali Vishnu Kute

Email ID: [sayalikute2507@gmail.com](mailto:sayalikute2507@gmail.com)

The screenshot shows the Android Studio interface with the Earthquake.java file open in the editor. The code implements a service that reads earthquake data from a file and handles menu and dialog interactions. The Java code is as follows:

```
// add the new earthquake entry to the list.
earthquakeList.add(quake);
// end of while loop
// end of reading all the entries return the list to onPostExecute.
return earthquakeList;
} catch (Exception e) {
    e.printStackTrace();
}
return null;
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    super.onCreateOptionsMenu(menu);
    menu.add(0, MENU_UPDATE, Menu.NONE, "Refresh Earthquakes");
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    super.onOptionsItemSelected(item);
    switch (item.getItemId()) {
        case (MENU_UPDATE):
            refreshEarthquakes();
            return true;
    }
    return false;
}

@Override
public Dialog onCreateDialog(int id) {
    switch(id) {
        case (QUAKE_DIALOG):
            LayoutInflator li = LayoutInflater.from(this);

```

The Android Monitor and Terminal tabs are visible at the bottom.

The screenshot shows the Android Studio interface with the Earthquake.java file open in the editor. The code implements a service that reads earthquake data from a file and handles menu and dialog interactions. The Java code is as follows:

```
@Override
public Dialog onCreateDialog(int id) {
    switch(id) {
        case (QUAKE_DIALOG):
            LayoutInflator li = LayoutInflater.from(this);
            View quakeDetailsView = li.inflate(R.layout.quake_details, null);

            AlertDialog.Builder quakeDialog = new AlertDialog.Builder(this);
            quakeDialog.setTitle("Quake Time");
            quakeDialog.setView(quakeDetailsView);
            return quakeDialog.create();
    }
    return null;
}

@Override
public void onPrepareDialog(int id, Dialog dialog) {
    switch(id) {
        case (QUAKE_DIALOG):
            SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
            String dateString = sdf.format(selectedQuake.getDate());
            String quakeText = "Magnitude " + selectedQuake.getMagnitude() +
                "\n" + selectedQuake.getDetails() + "\n" +
                selectedQuake.getLink();

            AlertDialog quakeDialog = (AlertDialog)dialog;
            quakeDialog.setTitle(dateString);
            TextView tv = (TextView)quakeDialog.findViewById(R.id.quakeDetailsTextView);
            tv.setText(quakeText);
            break;
    }
}

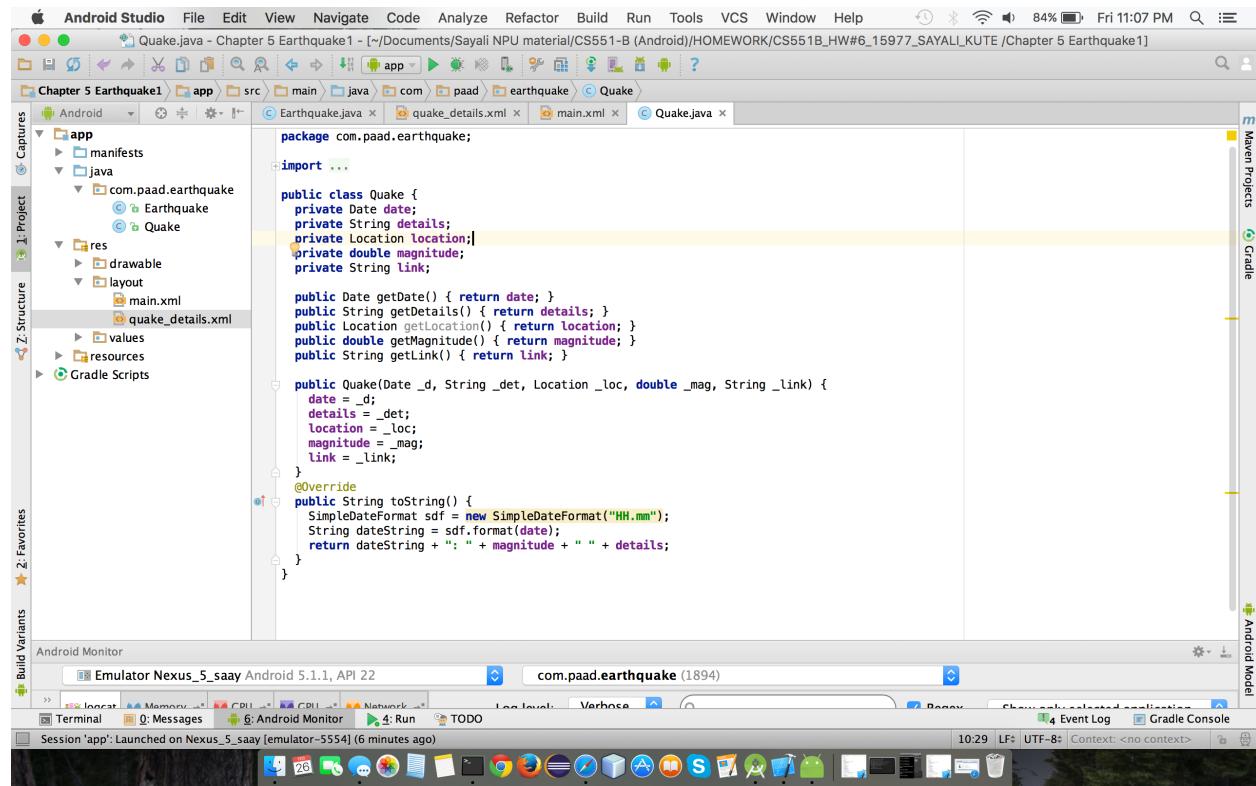
```

The Android Monitor and Terminal tabs are visible at the bottom.

Name: Sayali Vishnu Kute

Email ID: [sayalikute2507@gmail.com](mailto:sayalikute2507@gmail.com)

## Create Quake.java class as follows:



```
package com.paad.earthquake;

import ...

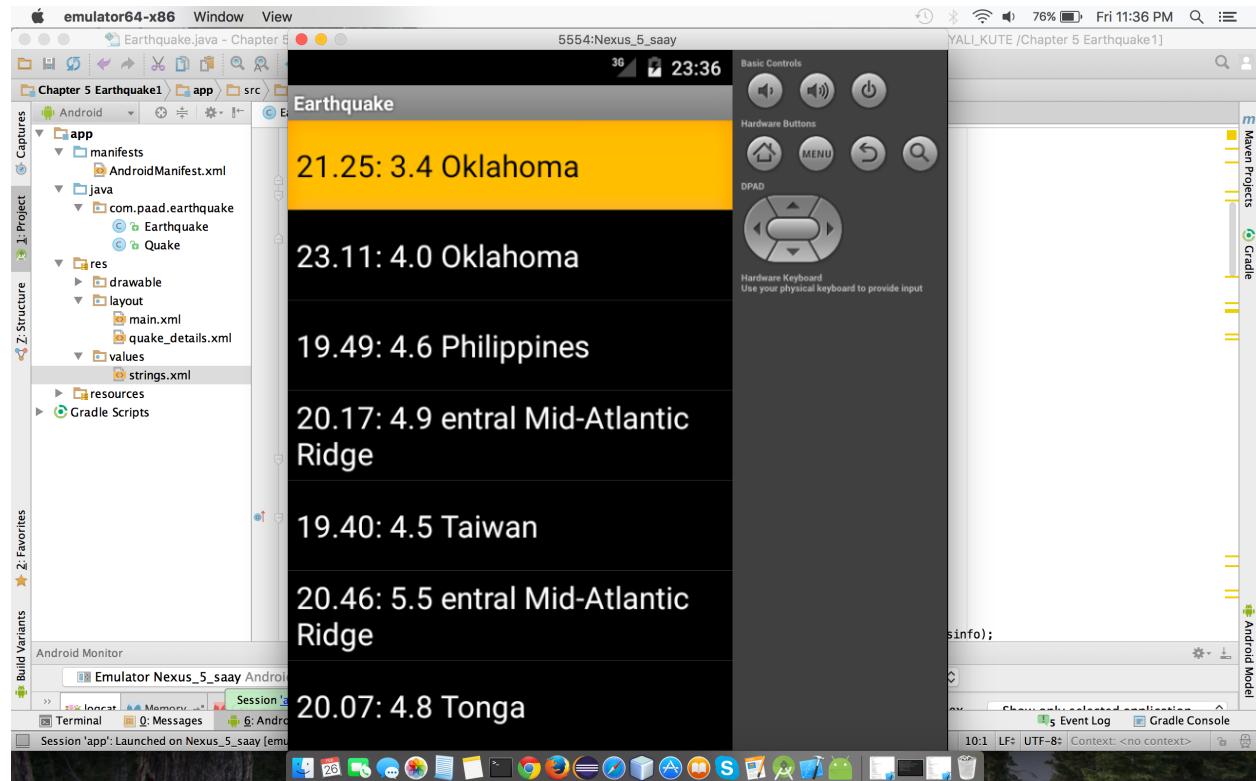
public class Quake {
    private Date date;
    private String details;
    private Location location;
    private double magnitude;
    private String link;

    public Date getDate() { return date; }
    public String getDetails() { return details; }
    public Location getLocation() { return location; }
    public double getMagnitude() { return magnitude; }
    public String getLink() { return link; }

    public Quake(Date _d, String _det, Location _loc, double _mag, String _link) {
        date = _d;
        details = _det;
        location = _loc;
        magnitude = _mag;
        link = _link;
    }

    @Override
    public String toString() {
        SimpleDateFormat sdf = new SimpleDateFormat("HH:mm");
        String dateString = sdf.format(date);
        return dateString + ":" + magnitude + " " + details;
    }
}
```

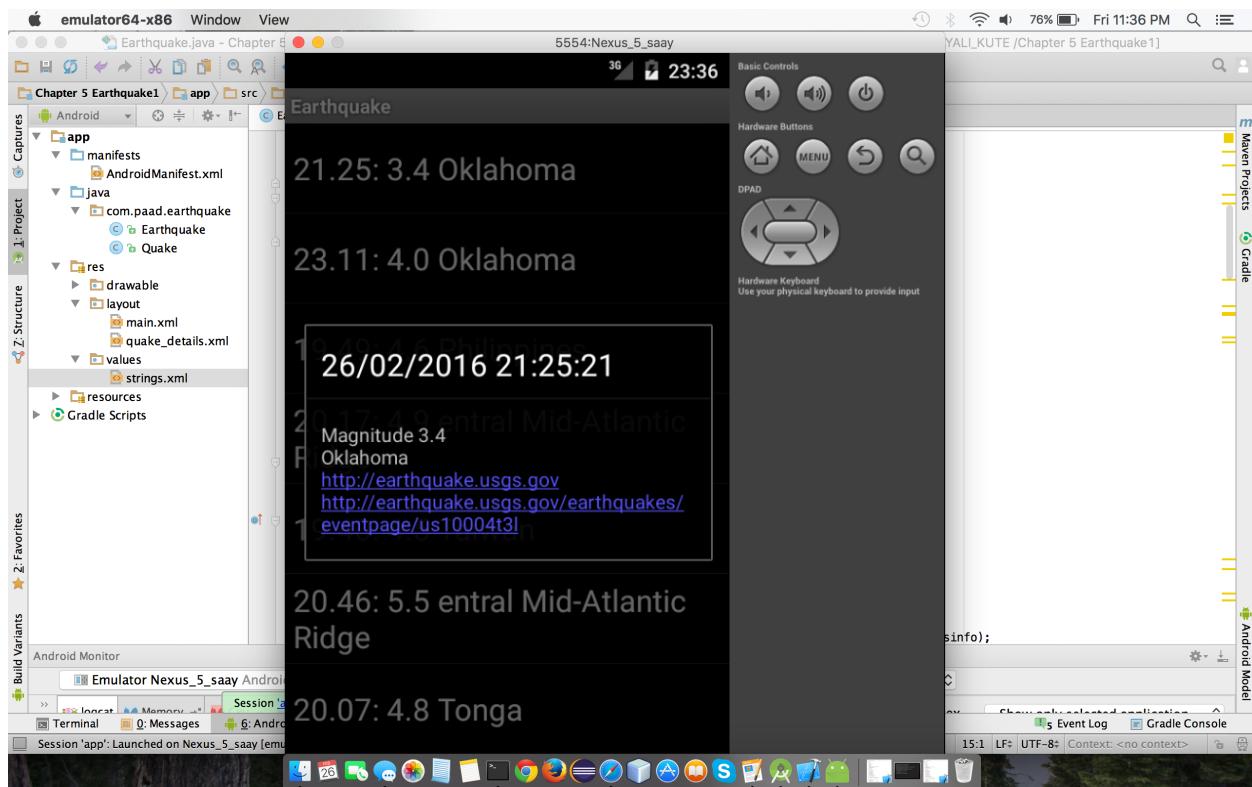
Run the program on Android Studio Emulator:  
A list of earthquakes will appear:



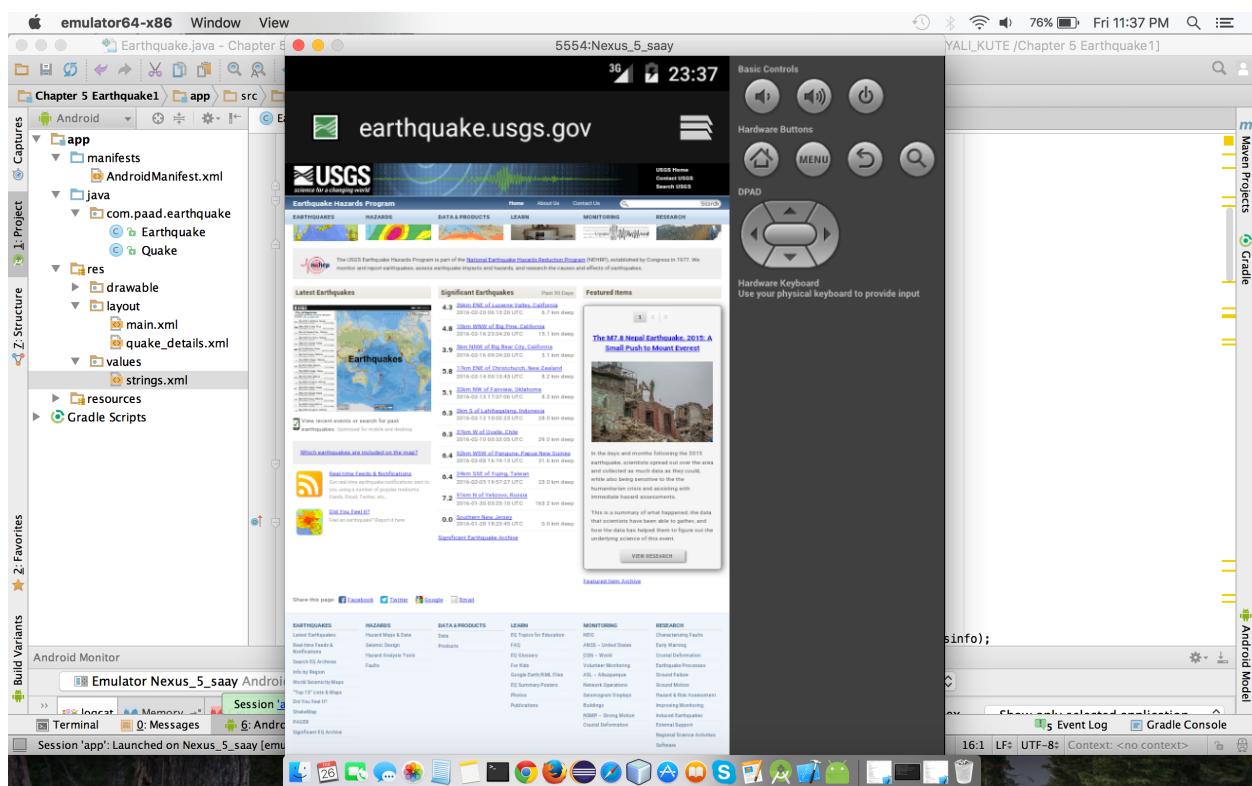
Name: Sayali Vishnu Kute

Email ID: [sayalikute2507@gmail.com](mailto:sayalikute2507@gmail.com)

Select one of the earthquakes from the list. Details of that earthquake like date, time, location, magnitude, link will appear:



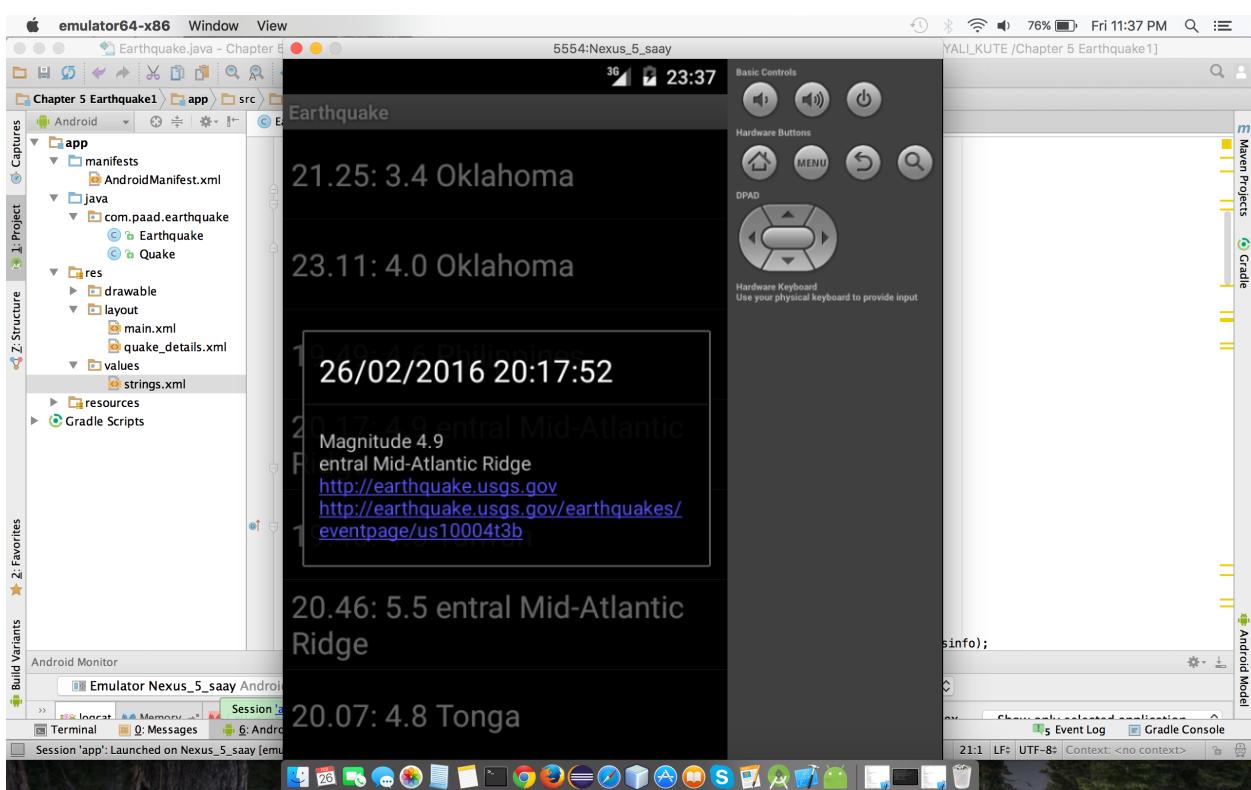
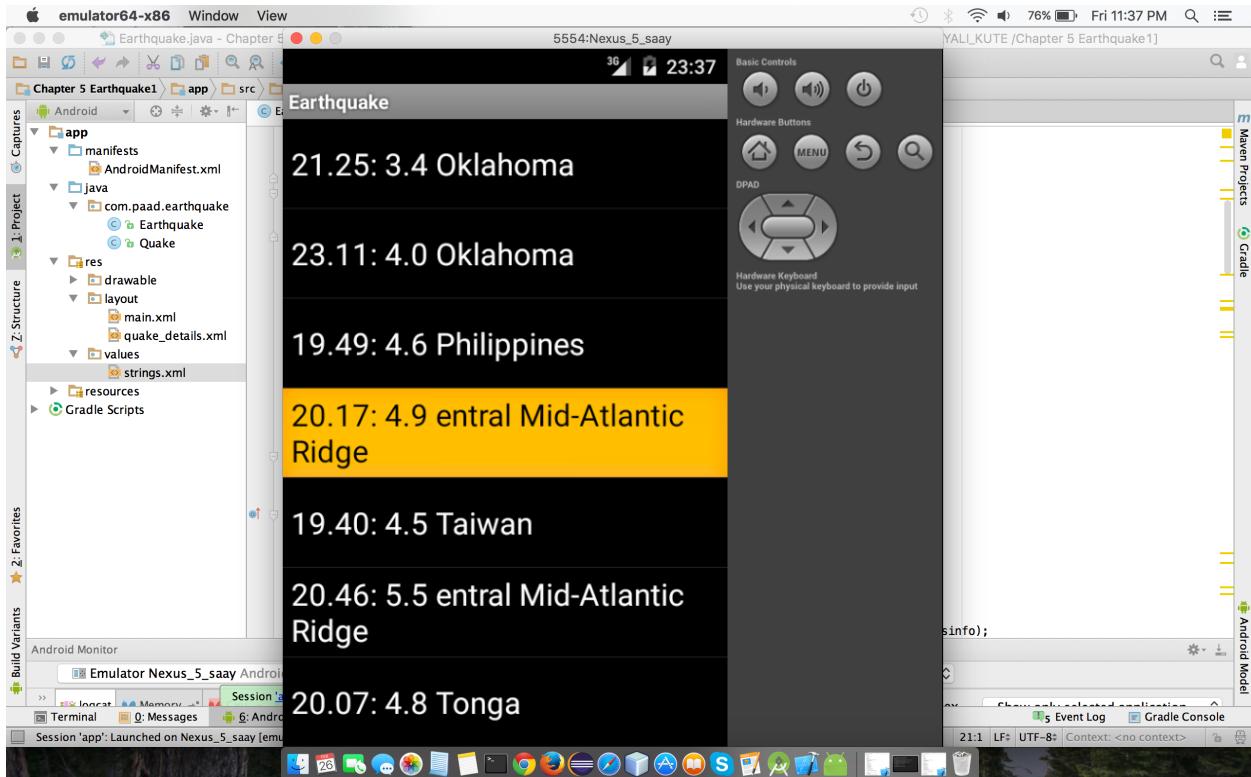
The earthquake link will take you to the [earthquake.usgs.gov](http://earthquake.usgs.gov) website:



Name: Sayali Vishnu Kute

Email ID: [sayalikute2507@gmail.com](mailto:sayalikute2507@gmail.com)

## Same steps are followed for other earthquakes:



Name: Sayali Vishnu Kute

Email ID: [sayalikute2507@gmail.com](mailto:sayalikute2507@gmail.com)

