EnergyFlo

CIS 470 | Final Project

# Idea

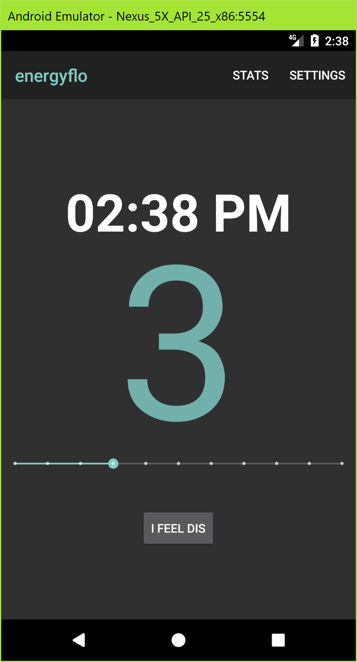
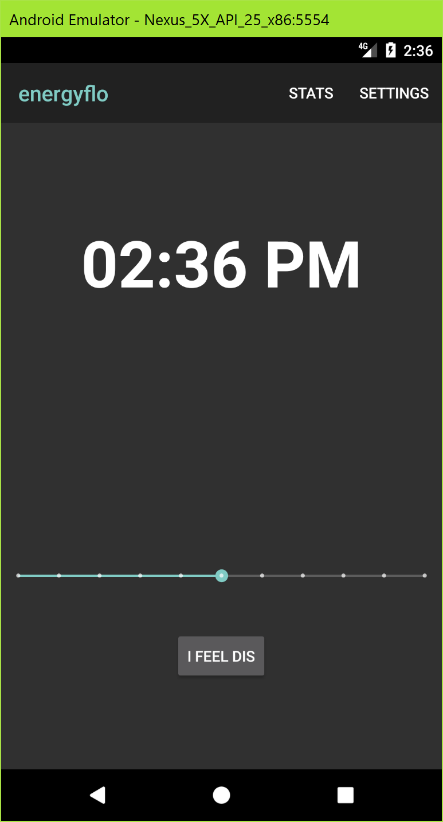
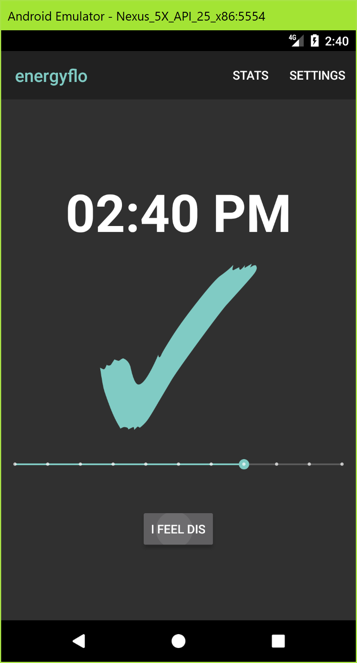
Energyflo is a personal-productivity optimizer.

EnergyFlo allows its user to record their energy level throughout the day, and provides automatic analysis and graphing of user data, helping the user to identify their hours of peak-energy. This in turn allows the user to prioritize their most important work for their most productive hours of the day.

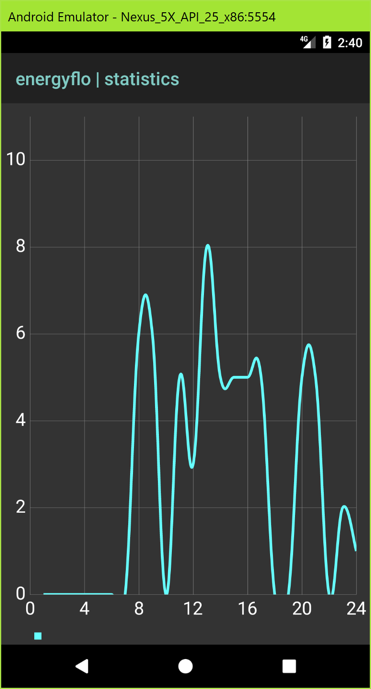
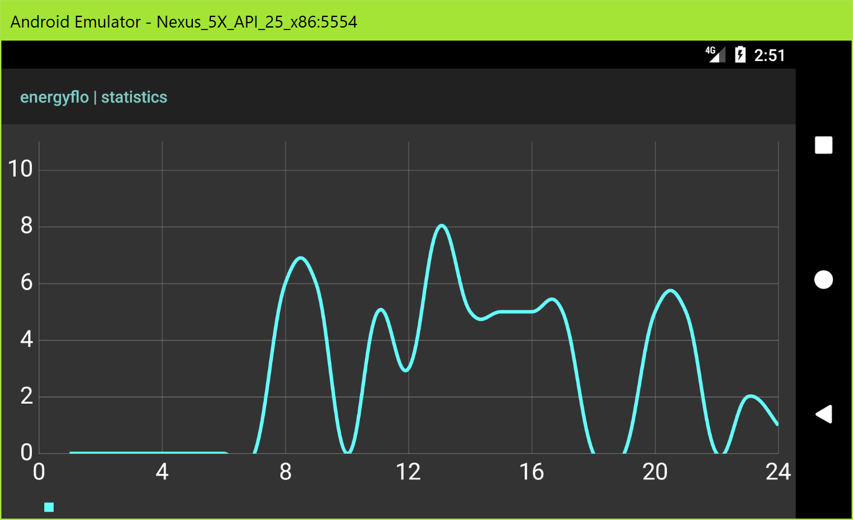
# UI Design

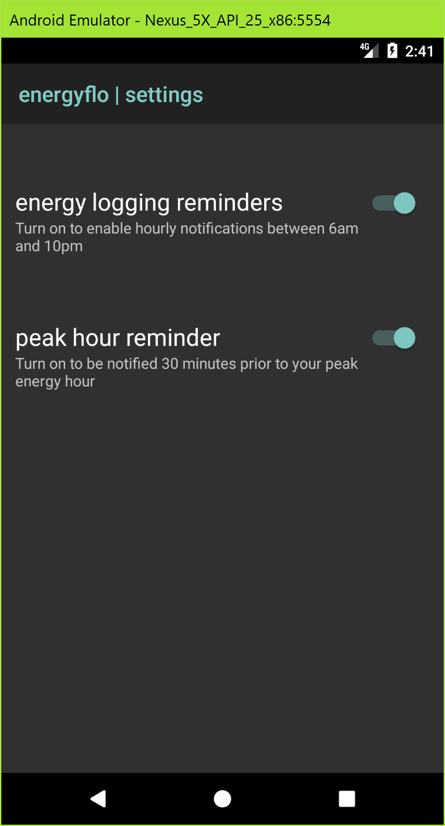
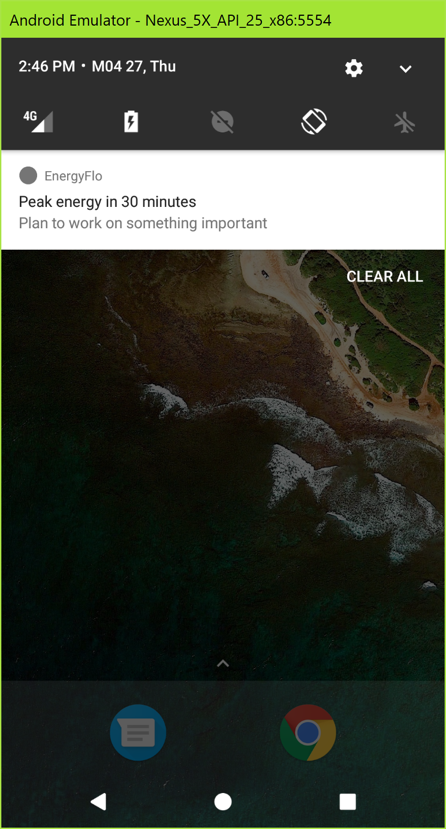
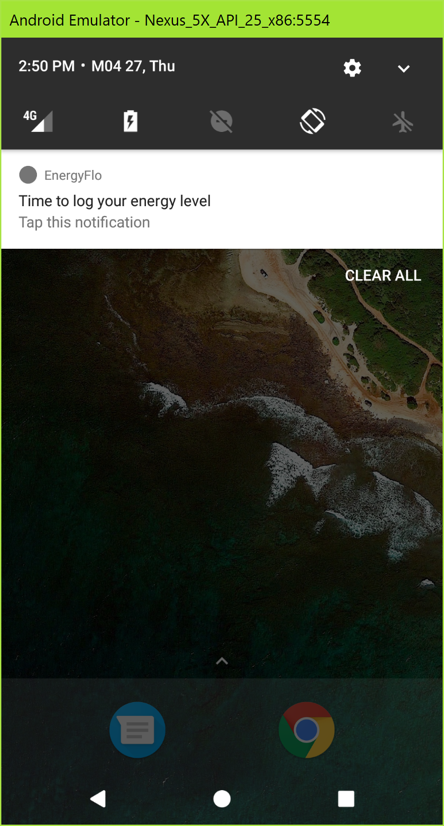
EnergyFlo is designed to be simple and clean. The user is presented with the current time and a seekbar when the app is first loaded.

When moving the seek bar, the user is presented with a number from 1 to 10 above to show what level they are selecting. After a moment, the number fades away. When the user clicks the “I FEEL DIS” button, the app flashes a check mark to let them know their data was stored.

EnergyFlo’s toolbar contains two buttons: STATS and SETTINGS. When STATS is selected, the user is taken to a graph showing all twenty four hours of the day and what their average recorded energy level is for each hour. The SETTINGS button takes the user to an interface where they can toggle on or off the provided notifications options. The first option allows the app to remind the user to log their energy level at hourly intervals. The second option notifies the user thirty minutes before their hour of peak energy (which is the hour with the highest average energy rating).

# Technical Design:

When the app starts, a database is created if it doesn’t already exist. Then all the user has to do is set the rating bar to their current energy level and click the button to record their rating. EnergyFlo uses an SQLite database. The database handler **DBHandler.java** uses the (poorly named) **Log** class to store, pull, and update the database entries.

Alarms are handled in two java classes. First, **SettingsActivity.java** handles user interaction with the switches on the Settings activity, and schedules/cancels alarms with the Android system appropriately. The pending intents for these alarms are received by **AlarmReciever.java**, which extends BroadcastReceiver. This class uses Notification.Builder and NotificationManager to build and send the appropriate notification to Android’s notification service.

Graphs are created and displayed in our **StatsActivity.java** activity. The [MPAndroidChart](https://github.com/PhilJay/MPAndroidChart) library was used for drawing the graph. The program loops through all values in the database and plots the user’s average energy rating for each hour of the day.

# Acknowledgements

## Graph

**MPAndroidChart** by PhilJay

(<https://github.com/PhilJay/MPAndroidChart>)

## AlarmManager

Tutorial by Learn Android Easily.com

(<http://www.learn-android-easily.com/2013/06/scheduling-task-using-alarm-manager.html>)

## General

StackOverflow

(<http://www.stackoverflow.com/>)

Android Developers

(<https://developer.android.com/index.html>)

# Code

## MainActivity.java

/\*  
\* GROUP 16  
\* WILLIAM PIERCE 2643343  
\* NAM DO 2594704  
\* DILLON PURVIS 2532954  
\* \*/  
  
package com.example.billy.energyflo;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.support.v7.app.AppCompatActivity;  
import android.support.v7.widget.Toolbar;  
import android.view.Menu;  
import android.view.MenuItem;  
import android.view.View;  
import android.view.animation.AlphaAnimation;  
import android.view.animation.Animation;  
import android.widget.SeekBar;  
import android.widget.TextClock;  
import android.widget.TextView;  
  
public class MainActivity extends AppCompatActivity {  
  
 DBHandler mDbHelper;  
 TextClock time;  
 SeekBar ratingSelector;  
 TextView ratingTextView;  
 Animation out;  
  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 Toolbar myToolbar = (Toolbar) findViewById(R.id.*toolbar*);  
 setSupportActionBar(myToolbar);  
  
 ratingTextView = (TextView) findViewById(R.id.*ratingText*);  
 mDbHelper = new DBHandler(this.getApplicationContext());  
 time = (TextClock) findViewById(R.id.*textClock*);  
 ratingSelector = (SeekBar) findViewById(R.id.*seekBar*);  
 ratingSelector.setOnSeekBarChangeListener(yourListener);  
  
// getSupportActionBar().setDisplayHomeAsUpEnabled(true);  
//  
// getSupportActionBar().setHomeButtonEnabled(true);  
  
 out = new AlphaAnimation(1.0f, 0.0f);  
 out.setDuration(3000);  
 out.setAnimationListener(new Animation.AnimationListener() {  
  
 @Override  
 public void onAnimationStart(Animation animation) {  
  
 }  
  
 @Override  
 public void onAnimationEnd(Animation animation) {  
 ratingTextView.setText("");  
 //mSwitcher.startAnimation(in);  
  
 }  
  
 @Override  
 public void onAnimationRepeat(Animation animation) {  
  
 }  
 });  
  
 }  
  
  
 private SeekBar.OnSeekBarChangeListener yourListener = new SeekBar.OnSeekBarChangeListener() {  
  
 @Override  
 public void onProgressChanged(SeekBar seekBar, int progress,  
 boolean fromUser) {  
 // Log the progress  
 android.util.Log.*d*("DEBUG", "Progress is: "+progress);  
 //set ratingTextView's text  
 ratingTextView.setText(""+progress);  
 ratingTextView.startAnimation(out);  
 }  
  
 public void onStartTrackingTouch(SeekBar seekBar) {}  
  
 public void onStopTrackingTouch(SeekBar seekBar) {}  
  
 };  
  
  
  
 public void addItem(View view){  
  
 //*TODO give user feedback that their button press was recorded* /\*Determine Current Time and Parse\*/  
  
 CharSequence currentTimeWholeString = time.getText();  
 android.util.Log.*d*("addItem", "currentTimeWholeString: " + time.getText());  
  
  
 CharSequence amPm = currentTimeWholeString.subSequence(6,8);  
 CharSequence currentTime = currentTimeWholeString.subSequence(0,2);  
 int number\_time = Integer.*parseInt*(currentTime.toString());  
  
 android.util.Log.*d*("addItem", "amPm: " + amPm);  
 android.util.Log.*d*("addItem", "currentTime: " + currentTime);  
 android.util.Log.*d*("addItem", "number\_time: " + number\_time);  
  
 if((amPm.toString().equals("PM") && !currentTime.toString().equals("12")) || ((amPm.toString().equals("AM") && currentTime.toString().equals("12")))){  
 // if it is afternoon, or, if it is midnight: convert to military time equivalent  
 number\_time += 12;  
 //android.util.Log.d("addItem", "IT IS NIGHTTIME add 12 = " + number\_time);  
 }  
  
 android.util.Log.*d*("addItem", "number\_time (after condition): " + number\_time);  
  
 /\*Fetch User Rating\*/  
 int number\_rating = ratingSelector.getProgress();  
 android.util.Log.*d*("addItem", "number\_rating: " + number\_rating);  
  
  
  
  
 try{  
 // Find EnergyLog for the current hour  
 Log current\_log = mDbHelper.getLog(number\_time);  
  
 //edit log to add new params and update  
 current\_log.updateLog(number\_rating);  
 mDbHelper.updateHour(current\_log);  
 android.util.Log.*d*("Success", "updated hour " + current\_log.getHour() +" Average = "+ current\_log.getAverage() + " Number of ratings = "+current\_log.getNumber\_of\_ratings() +" Total = " + current\_log.getTotal());  
  
 ratingTextView.setText("✓");  
 ratingTextView.startAnimation(out);  
 }  
 catch(Exception e){  
 android.util.Log.*d*("Exception", "Creating new EnergyLog for hour: " + number\_time);  
 Log log = new Log(number\_time,number\_rating,1,number\_rating);  
 mDbHelper.addHour(log);  
 return;  
 }  
  
 //android.util.Log.d("BAD", "Added value but didn't return");  
  
 }  
  
 @Override  
 public boolean onCreateOptionsMenu(Menu menu) {  
 getMenuInflater().inflate(R.menu.*mainactivitymenu*,menu);  
 return true;  
 }  
  
 @Override  
 public boolean onOptionsItemSelected(MenuItem item) {  
 if(item.getItemId()==R.id.*gotoStats*){  
 Intent intentToViewStats = new Intent(this, StatsActivity.class);  
 startActivity(intentToViewStats);  
 return true;  
 }  
 if (item.getItemId() == R.id.*gotoSettings*) {  
 Intent intentToViewSettings = new Intent(this, SettingsActivity.class);  
 startActivity(intentToViewSettings);  
 return true;  
  
 }  
  
  
 return super.onOptionsItemSelected(item);  
 }  
}

## DBHandler.java

package com.example.billy.energyflo;  
  
import android.content.ContentValues;  
import android.content.Context;  
import android.database.Cursor;  
import android.database.sqlite.SQLiteDatabase;  
import android.database.sqlite.SQLiteOpenHelper;  
  
*/\*\*  
 \* Created by Billy on 2/15/17.  
 \*/*public class DBHandler extends SQLiteOpenHelper {  
 // Database Version  
 private static final int *DATABASE\_VERSION* = 1;  
 // Database Name  
 private static final String *DATABASE\_NAME* = "energyflowdb";  
 //tables  
 private static final String *TABLE\_LOG* = "Log";  
 // Table Column names  
 private static final String *KEY\_HR* = "Hour";  
 private static final String *KEY\_AVG* = "Average";  
 private static final String *KEY\_NUM\_OF\_RATINGS* = "Number\_Of\_Ratings";  
 private static final String *KEY\_TOTAL* = "Total";  
 private static final String *CREATE\_LOG\_TABLE* = "CREATE TABLE " + *TABLE\_LOG* + "("  
 + *KEY\_HR* + " INTEGER PRIMARY KEY," + *KEY\_AVG* + " DOUBLE,"  
 + *KEY\_NUM\_OF\_RATINGS* + " INTEGER," + *KEY\_TOTAL* + " INTEGER)";  
 //DBHandler mDbHelper = new DBHandler(getContext());  
  
  
  
 public DBHandler(Context context) {  
 super(context, *DATABASE\_NAME*, null, *DATABASE\_VERSION*);  
 }  
  
 private static DBHandler *mInstance* = null;  
  
 public static DBHandler getInstance(Context activityContext) {  
  
 // Get the application context from the activityContext to prevent leak  
 if (*mInstance* == null) {  
 *mInstance* = new DBHandler (activityContext.getApplicationContext());  
 }  
 return *mInstance*;  
 }  
  
 @Override  
 public void onCreate(SQLiteDatabase db) {  
 db.execSQL(*CREATE\_LOG\_TABLE*);  
  
 }  
  
 @Override  
 public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  
  
 db.execSQL("DROP TABLE IF EXISTS " + *TABLE\_LOG*);  
// Creating tables again  
 onCreate(db);  
 }  
  
 // Adding new hour  
 public void addHour(Log log) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 ContentValues values = new ContentValues();  
 values.put(*KEY\_HR*, log.getHour());  
 values.put(*KEY\_AVG*, log.getAverage());  
 values.put(*KEY\_NUM\_OF\_RATINGS*, log.getNumber\_of\_ratings());  
 values.put(*KEY\_TOTAL*,log.getTotal());  
  
// Inserting Row  
 try{  
  
 long insert = db.insert(*TABLE\_LOG*, null, values);  
 android.util.Log.*v*("!!!!!!!!!!!", "insert = "+insert);  
 }  
 catch(Exception e){  
 android.util.Log.*v*("Entry Exists", "This is good");  
 }  
  
 db.close(); // Closing database connection  
  
 }  
 // Getting a row  
 public Log getLog(int id) {  
 SQLiteDatabase db = this.getReadableDatabase();  
 Cursor cursor = db.query(*TABLE\_LOG*, new String[] { *KEY\_HR*,  
 *KEY\_AVG*, *KEY\_NUM\_OF\_RATINGS*,*KEY\_TOTAL*}, *KEY\_HR* + "=?",  
 new String[] { String.*valueOf*(id) }, null, null, null, null);  
 if (cursor != null)  
 cursor.moveToFirst();  
 Log current = new Log(Integer.*parseInt*(cursor.getString(0)),  
 Double.*parseDouble*(cursor.getString(1)), Integer.*parseInt*(cursor.getString(2)),  
 Integer.*parseInt*(cursor.getString(3)));  
// return log  
 return current;  
 }  
 //update database  
 public void updateHour(Log log) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 ContentValues values = new ContentValues();  
 values.put(*KEY\_HR*, log.getHour());  
 values.put(*KEY\_AVG*, log.getAverage());  
 values.put(*KEY\_NUM\_OF\_RATINGS*, log.getNumber\_of\_ratings());  
 values.put(*KEY\_TOTAL*,log.getTotal());  
  
// Inserting Row  
 try{  
  
 long insert = db.update(*TABLE\_LOG*, values, *KEY\_HR* + " = ?",  
 new String[]{String.*valueOf*(log.getHour())});  
 android.util.Log.*v*("Update", "hour updated = "+ log.getHour());  
 }  
 catch(Exception e){  
 android.util.Log.*v*("Entry Exists", "This is good");  
 }  
  
 db.close(); // Closing database connection  
  
 }  
 public int findPeakHour(){  
 SQLiteDatabase db = this.getWritableDatabase();  
 Cursor cursor = db.rawQuery("SELECT MAX(average),Hour FROM log", null);  
 if (cursor != null)  
 cursor.moveToFirst();  
 android.util.Log.*d*("Highest Entry", String.*valueOf*(Double.*parseDouble*(cursor.getString(0)))+ " " + String.*valueOf*(Integer.*parseInt*(cursor.getString(1))));  
 return Integer.*parseInt*(cursor.getString(1));  
 }  
}

## Log.java

package com.example.billy.energyflo;  
  
*/\*\*  
 \* Created by Billy on 2/15/17.  
 \*/*public class Log {  
 private int hour;  
 private double average;  
 private int number\_of\_ratings;  
 private int total;  
 public Log()  
 {  
 }  
  
 public Log(int hour,double average,int number\_of\_ratings, int total)  
 {  
 this.hour=hour;  
 this.average=average;  
 this.number\_of\_ratings = number\_of\_ratings;  
 this.total = total;  
 }  
 public void setHour(int hour) {  
 this.hour = hour;  
 }  
 public void setAverage(Float avg) {  
 this.average = avg;  
 }  
 public void setNumber\_of\_ratings(int num) {  
 this.number\_of\_ratings = num;  
 }  
 public void setTotal(int tot){ this.total = tot; };  
 public int getHour() {  
 return this.hour;  
 }  
 public double getAverage() {  
 return this.average;  
 }  
 public int getNumber\_of\_ratings() {  
 return this.number\_of\_ratings;  
 }  
 public int getTotal(){ return this.total; }  
 public void updateLog(int rating){  
 //method updates the log that was grabbed from the database  
 //NOTE this does not update the database!  
 this.total += rating;  
 this.number\_of\_ratings++;  
 this.average = this.total/this.number\_of\_ratings;  
 }  
  
  
}

## SettingsActivity.java

package com.example.billy.energyflo;  
  
import android.app.AlarmManager;  
import android.app.PendingIntent;  
import android.content.Context;  
import android.content.Intent;  
import android.content.SharedPreferences;  
import android.icu.util.Calendar;  
import android.os.Bundle;  
import android.support.v7.app.AppCompatActivity;  
import android.util.Log;  
import android.widget.CompoundButton;  
import android.widget.Switch;  
  
public class SettingsActivity extends AppCompatActivity  
{  
 Switch reminderNotificationSwitch;  
 Switch peakNotificationSwitch;  
  
 @Override  
 public void onCreate(Bundle savedInstanceState)  
 {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_settings*);  
  
 /\* Retrieve user preferences \*/  
 SharedPreferences prefs = getSharedPreferences("com.example.billy.energyflo", *MODE\_PRIVATE*);  
 boolean logReminderSwitchState = prefs.getBoolean("logReminderSwitch", false);  
 boolean peakReminderSwitchState = prefs.getBoolean("peakReminderSwitch", false);  
  
  
 /\* Restore user preferences to the switches\*/  
 reminderNotificationSwitch = (Switch) findViewById(R.id.*notificationSwitch*);  
 reminderNotificationSwitch.setChecked(logReminderSwitchState);  
 peakNotificationSwitch = (Switch) findViewById(R.id.*peakNotificationSwitch*);  
 peakNotificationSwitch.setChecked(peakReminderSwitchState);  
  
  
 /\* Listen for changes to switch states \*/  
 reminderNotificationSwitch.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener() {  
 public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {  
  
 /\* Save state of switch when flipped \*/  
 SharedPreferences.Editor editor = getSharedPreferences("com.example.billy.energyflo", *MODE\_PRIVATE*).edit();  
 editor.putBoolean("logReminderSwitch", reminderNotificationSwitch.isChecked());  
 editor.commit();  
  
 if (isChecked) {  
 scheduleEnergyLoggingAlarm();  
 }  
 else {  
 cancelEnergyLoggingAlarm();  
 }  
 }  
 });  
  
  
 peakNotificationSwitch.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener() {  
 public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {  
  
 /\* Save state of switch when flipped \*/  
 SharedPreferences.Editor editor = getSharedPreferences("com.example.billy.energyflo", *MODE\_PRIVATE*).edit();  
 editor.putBoolean("peakReminderSwitch", peakNotificationSwitch.isChecked());  
 editor.commit();  
  
 if (isChecked) {  
 schedulePeakAlarm();  
 }  
 else {  
 cancelPeakAlarm();  
 }  
 }  
 });  
  
  
  
 }  
  
 public void scheduleEnergyLoggingAlarm()  
 {  
 Calendar calendar = Calendar.*getInstance*();  
 calendar.setTimeInMillis(System.*currentTimeMillis*());  
 AlarmManager alarmManager = (AlarmManager) getSystemService(Context.*ALARM\_SERVICE*);  
 Intent intentAlarm = new Intent(this, AlarmReceiver.class);  
 intentAlarm.putExtra("alarmType", "RECORDING\_REMINDER");  
  
  
  
 /\* Schedules hourly alarms between 6am and 10pm \*/  
 for (int reminderHour = 6; reminderHour < 23; reminderHour++) {  
  
 calendar.set(Calendar.*HOUR\_OF\_DAY*, reminderHour);  
  
 alarmManager.setInexactRepeating(AlarmManager.*RTC\_WAKEUP*,  
 calendar.getTimeInMillis(),  
 AlarmManager.*INTERVAL\_DAY*,  
 PendingIntent.*getBroadcast*(this,reminderHour, intentAlarm, PendingIntent.*FLAG\_UPDATE\_CURRENT*));  
  
 android.util.Log.*d*("SettingsActivity",  
 "Energy logging reminder #" + reminderHour  
 + " set for: " + calendar.getTimeInMillis());  
 }  
 }  
  
  
 public void cancelEnergyLoggingAlarm()  
 {  
 /\* Cancels hourly alarms between 6am and 10pm \*/  
 Intent intentAlarm = new Intent(this, AlarmReceiver.class);  
 AlarmManager alarmManager = (AlarmManager) getSystemService(Context.*ALARM\_SERVICE*);  
  
 for (int reminderHour = 6; reminderHour < 23; reminderHour++) {  
 PendingIntent pendingIntent = PendingIntent.*getBroadcast*(this,  
 reminderHour,  
 intentAlarm,  
 PendingIntent.*FLAG\_UPDATE\_CURRENT*);  
 alarmManager.cancel(pendingIntent);  
 }  
  
  
 Log.*d*("SettingsActivity", "Cancelled energy logging alarms");  
 }  
  
  
  
  
 public void schedulePeakAlarm() {  
 DBHandler dbHandler = new DBHandler(this);  
 int peakHour = dbHandler.findPeakHour();  
  
 Calendar calendar = Calendar.*getInstance*();  
 calendar.setTimeInMillis(System.*currentTimeMillis*());  
// calendar.setTimeInMillis(System.currentTimeMillis() + 1801000L); // for immediate testing of notification  
 calendar.set(Calendar.*HOUR\_OF\_DAY*, peakHour);  
 android.util.Log.*d*("Settings Activity", "set alarm for peak hour: " + String.*valueOf*(calendar.getTimeInMillis()));  
  
 Intent intentAlarm = new Intent(this, AlarmReceiver.class);  
 intentAlarm.putExtra("alarmType", "PEAK\_REMINDER");  
  
 AlarmManager alarmManager = (AlarmManager) getSystemService(Context.*ALARM\_SERVICE*);  
  
 // subtract 1800000 ms in order to make alarm 30 minutes before peak hour  
 //*TODO add a scheduled daily task to update alarm when peak hour changes* alarmManager.setRepeating(AlarmManager.*RTC\_WAKEUP*, calendar.getTimeInMillis() - 1800000L,  
 AlarmManager.*INTERVAL\_DAY*,  
 PendingIntent.*getBroadcast*(this, 2, intentAlarm, PendingIntent.*FLAG\_UPDATE\_CURRENT*));  
 }  
  
  
 public void cancelPeakAlarm()  
 {  
 Intent intentAlarm = new Intent(this, AlarmReceiver.class);  
 PendingIntent pendingIntent = PendingIntent.*getBroadcast*(this, 2, intentAlarm, PendingIntent.*FLAG\_UPDATE\_CURRENT*);  
 AlarmManager alarmManager = (AlarmManager) getSystemService(Context.*ALARM\_SERVICE*);  
  
 alarmManager.cancel(pendingIntent);  
  
 Log.*d*("SettingsActivity", "Cancelled peak energy alarm");  
 }  
  
}

## AlarmReceiver.java

package com.example.billy.energyflo;  
  
import android.app.Notification;  
import android.app.NotificationManager;  
import android.app.PendingIntent;  
import android.content.BroadcastReceiver;  
import android.content.Context;  
import android.content.Intent;  
  
  
public class AlarmReceiver extends BroadcastReceiver  
{  
 @Override  
 public void onReceive(Context context, Intent intent)  
 {  
 android.util.Log.*d*("AlarmReceiver", "received alarm broadcast");  
  
 String alarmType = intent.getStringExtra("alarmType");  
  
 if (alarmType.equals("RECORDING\_REMINDER")) {  
  
 Notification.Builder myBuilder = new Notification.Builder(context)  
 .setContentTitle("Time to log your energy level")  
 .setContentText("Tap this notification")  
 .setSmallIcon(R.mipmap.*ic\_launcher\_textver*)  
 .setAutoCancel(true);  
  
 Intent myIntent = new Intent(context, MainActivity.class);  
  
 PendingIntent myPendingIntent = PendingIntent.*getActivity*(context, 0 , myIntent, 0);  
 myBuilder.setContentIntent(myPendingIntent);  
  
 Notification myNotification = myBuilder.build();  
 NotificationManager myNotificationManager =  
 (NotificationManager) context.getSystemService(Context.*NOTIFICATION\_SERVICE*);  
 myNotificationManager.notify(1234, myNotification);  
  
 }  
  
  
 if (alarmType.equals("PEAK\_REMINDER")) {  
 Notification.Builder myBuilder = new Notification.Builder(context)  
 .setContentTitle("Peak energy in 30 minutes")  
 .setContentText("Plan to work on something important")  
 .setSmallIcon(R.mipmap.*ic\_launcher\_textver*)  
 .setAutoCancel(true);  
  
 Intent myIntent = new Intent(context, MainActivity.class);  
  
 PendingIntent myPendingIntent = PendingIntent.*getActivity*(context, 0 , myIntent, 0);  
 myBuilder.setContentIntent(myPendingIntent);  
  
 Notification myNotification = myBuilder.build();  
 NotificationManager myNotificationManager =  
 (NotificationManager) context.getSystemService(Context.*NOTIFICATION\_SERVICE*);  
 myNotificationManager.notify(4567, myNotification);  
 }  
  
  
 }  
}

## StatsActivity.java

package com.example.billy.energyflo;  
  
import android.graphics.Color;  
import android.os.Bundle;  
import android.support.v7.app.AppCompatActivity;  
import android.support.v7.widget.Toolbar;  
  
import com.github.mikephil.charting.charts.LineChart;  
import com.github.mikephil.charting.components.Description;  
import com.github.mikephil.charting.components.Legend;  
import com.github.mikephil.charting.components.XAxis;  
import com.github.mikephil.charting.components.YAxis;  
import com.github.mikephil.charting.data.Entry;  
import com.github.mikephil.charting.data.LineData;  
import com.github.mikephil.charting.data.LineDataSet;  
import com.github.mikephil.charting.interfaces.datasets.ILineDataSet;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class StatsActivity extends AppCompatActivity {  
  
 DBHandler mDbHelper;  
 int hour;  
 Log thisHour;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_stats*);  
  
 Toolbar myToolbar = (Toolbar) findViewById(R.id.*statsToolbar*);  
 setSupportActionBar(myToolbar);  
  
 mDbHelper = new DBHandler(this.getApplicationContext());  
  
 android.util.Log.*d*("statsActivity", "Begin Logging from Stats Activity");  
 /\*  
 \*  
 \* Graph from: https://github.com/PhilJay/MPAndroidChart  
 \*  
 \*/  
  
 //Initialize a chart  
  
 LineChart chart = (LineChart) findViewById(R.id.*chart*);  
 chart.setBackgroundColor(Color.*rgb*(51, 51, 51));  
 Description des= chart.getDescription();  
 des.setEnabled(false);  
 Legend leg=chart.getLegend();  
 leg.setEnabled(true);  
 chart.setDrawGridBackground(false); //no grid line on background  
  
 //format X-axis  
 XAxis xAxis = chart.getXAxis();  
 xAxis.setPosition(XAxis.XAxisPosition.*BOTTOM*);  
 xAxis.setTextSize(20f);  
 xAxis.setTextColor(Color.*rgb*(255, 255, 255));  
 xAxis.setAxisMinValue(0);  
 xAxis.setAxisMaximum(24);  
  
 //format Y-axis  
 YAxis yAxis=chart.getAxisLeft();  
 yAxis.setTextSize(20f);  
 yAxis.setTextColor(Color.*rgb*(255, 255, 255));  
 yAxis.setAxisMinValue(0);  
 yAxis.setAxisMaximum(11);  
 chart.getAxisRight().setEnabled(false);  
  
 //Initialize a List of Entries to store all the data points  
 List<Entry> averageHourRating = new ArrayList<>();  
 //Loop through the database and add average rating on appropriate data points on line graph, in case the hour has not been recorded, rating is 0  
 //All the entries are stored in  
 for (hour=1; hour<=24;hour++){  
 try{  
 thisHour = mDbHelper.getLog(hour);  
 android.util.Log.*d*("StatsActivity","Hour looking at: " + thisHour.getHour() +" ,The AVERAGE IS: "+thisHour.getAverage());  
 Entry entry = new Entry(thisHour.getHour(),(float)thisHour.getAverage());  
 averageHourRating.add(entry);  
 }catch(Exception e){  
 Entry entry = new Entry(hour,0f);  
 averageHourRating.add(entry);  
 }  
 }  
  
 //Initialize the line with data by passing a list of data needed to be displayed  
 LineDataSet setHour= new LineDataSet(averageHourRating, " ");  
 //Formatting its elements (text, dot, line, color)  
 setHour.setDrawCircles(false);  
 setHour.setLineWidth(3f);  
 setHour.setColors(Color.*rgb*(102, 255, 255));  
 setHour.setAxisDependency(YAxis.AxisDependency.*LEFT*);  
 setHour.setMode(LineDataSet.Mode.*CUBIC\_BEZIER*); //instead of straight lines, draw line in cubic form  
 setHour.setCircleColor(Color.*rgb*(255, 255, 255));  
 setHour.setCircleRadius(4f);  
 setHour.setDrawCircleHole(false);  
 setHour.setValueTextSize(20f);  
 setHour.setValueTextColor(Color.*rgb*(102, 255, 255));  
 setHour.setDrawValues(false);  
  
 //Parse the data in appropriate data set for the Line chart class  
 List<ILineDataSet> dataSets= new ArrayList<>();  
 dataSets.add(setHour);  
  
 //Initialize the line chart class  
 LineData data= new LineData(dataSets);  
 chart.setData(data);  
 chart.invalidate(); //refreshing the chart every time the chart is brought up  
 }  
}

## activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>  
 <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/activity\_main"  
 style="@style/Theme.AppCompat.NoActionBar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_alignParentStart="false"  
 tools:context="com.example.billy.energyflo.MainActivity">  
  
 <android.support.v7.widget.Toolbar  
 android:id="@+id/toolbar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentStart="true"  
 android:layout\_alignParentTop="true"  
 android:background="?attr/colorPrimary"  
 android:minHeight="?attr/actionBarSize"  
 android:theme="?attr/actionBarTheme"  
 app:title="energyflo"  
 app:titleTextColor="?attr/colorAccent" />  
  
  
 <TextView  
 android:id="@+id/ratingText"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerHorizontal="true"  
 android:layout\_centerVertical="true"  
 android:textColor="?attr/colorAccent"  
 android:textSize="250sp" />  
  
 <TextClock  
 android:id="@+id/textClock"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@+id/toolbar"  
 android:layout\_centerHorizontal="true"  
 android:layout\_marginTop="90dp"  
 android:format12Hour="hh:mm a"  
 android:textSize="60sp"  
 android:textStyle="bold" />  
  
 <Button  
 android:id="@+id/button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentBottom="true"  
 android:layout\_centerHorizontal="true"  
 android:layout\_marginBottom="82dp"  
 android:elevation="0dp"  
 android:onClick="addItem"  
 android:text="@string/add\_db\_item" />  
  
 <SeekBar  
 android:id="@+id/seekBar"  
 style="@style/Widget.AppCompat.SeekBar.Discrete"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_above="@+id/button"  
 android:layout\_alignParentStart="true"  
 android:layout\_marginBottom="42dp"  
 android:max="10"  
 android:progress="5"  
 android:progressTint="?attr/colorAccent" />  
  
  
</RelativeLayout>

## activity\_settings.xml

<?xml version="1.0" encoding="utf-8"?>  
<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"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 tools:context="com.example.billy.energyflo.SettingsActivity">  
  
 <android.support.v7.widget.Toolbar  
 android:id="@+id/toolbar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentStart="true"  
 android:layout\_alignParentTop="true"  
 android:background="?attr/colorPrimary"  
 android:minHeight="?attr/actionBarSize"  
 android:theme="?attr/actionBarTheme"  
 app:title="energyflo | settings"  
 app:titleTextColor="?attr/colorAccent" />  
  
 <Switch  
 android:id="@+id/notificationSwitch"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignBaseline="@+id/textView3"  
 android:layout\_alignBottom="@+id/textView3"  
 android:layout\_alignParentEnd="true"  
 android:layout\_marginEnd="23dp"  
 android:splitTrack="false"  
 android:switchMinWidth="55dp"  
 android:text=""  
 android:textAllCaps="false"  
 android:textSize="18sp"  
 tools:layout\_editor\_absoluteX="149dp"  
 tools:layout\_editor\_absoluteY="250dp" />  
  
 <TextView  
 android:id="@+id/textView3"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentStart="true"  
 android:layout\_below="@+id/toolbar"  
 android:layout\_marginStart="13dp"  
 android:layout\_marginTop="57dp"  
 android:text="@string/energy\_logging\_reminders\_setting\_title"  
 android:textAppearance="@style/TextAppearance.AppCompat.Large" />  
  
 <TextView  
 android:id="@+id/textView4"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignStart="@+id/textView3"  
 android:layout\_below="@+id/textView3"  
 android:layout\_toStartOf="@+id/notificationSwitch"  
 android:text="@string/energy\_logging\_reminders\_setting\_description" />  
  
 <TextView  
 android:id="@+id/textView5"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignStart="@+id/textView4"  
 android:layout\_below="@+id/textView4"  
 android:layout\_marginTop="60dp"  
 android:text="@string/peak\_hour\_reminder\_setting\_title"  
 android:textAppearance="@style/TextAppearance.AppCompat.Large" />  
  
 <Switch  
 android:id="@+id/peakNotificationSwitch"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignBottom="@+id/textView5"  
 android:layout\_alignEnd="@+id/notificationSwitch"  
 android:switchMinWidth="55dp" />  
  
 <TextView  
 android:id="@+id/textView6"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignStart="@+id/textView5"  
 android:layout\_below="@+id/textView5"  
 android:text="@string/peak\_hour\_reminder\_setting\_description"  
 android:layout\_toStartOf="@+id/peakNotificationSwitch" />  
</RelativeLayout>

## activity\_stats.xml

<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context="com.example.billy.energyflo.StatsActivity">  
  
 <android.support.v7.widget.Toolbar  
 android:id="@+id/statsToolbar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentStart="true"  
 android:layout\_alignParentTop="true"  
 android:background="?attr/colorPrimary"  
 android:minHeight="?attr/actionBarSize"  
 android:theme="?attr/actionBarTheme"  
 app:title="energyflo | statistics"  
 app:titleTextColor="?attr/colorAccent" />  
  
 <com.github.mikephil.charting.charts.LineChart  
 android:id="@+id/chart"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_below="@+id/statsToolbar" />  
  
</RelativeLayout>