Smart Energy Recommender

(SmartER)

FIT 5046 Practical Assignment Phase 2 – 2018

(Semester I)

Student Name: Burak Seyhan – Laboratory 6

Student ID: 28867041

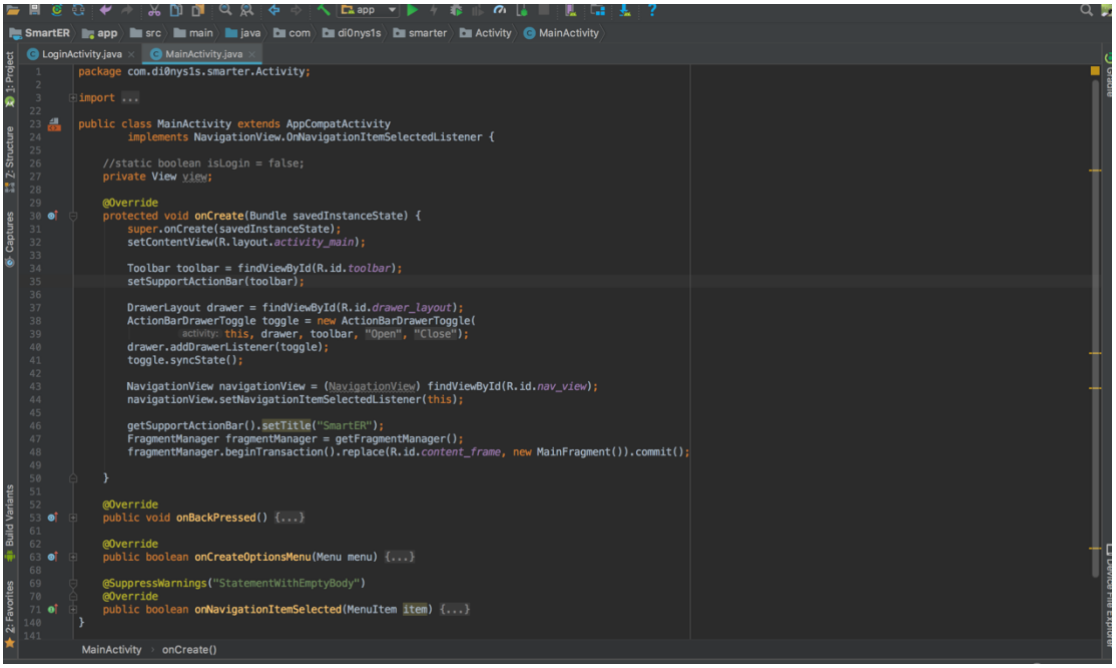
Lecturer: Pari Delil Haghighi
Tutor: Heaven Hu

TABLE OF CONTENTS

Smart Energy Recommender (SmartER) 1
FIT 5046 Practical Assignment Phase 2 – 2018 (Semester I) 1
Task 1 – Activity 3
1. Main Activity 3
2. Login Activity 3
3. Register Activity 4
..... 5
Task 2 – DB Manager and DB Structure 5
Task 3 – Fragments, Functions, Services and Classes 6
3.1 Fragments 6
3.1.1 Main Fragment 6
3.1.2 Map Fragment 7
3.1.3 Report Fragment 9
3.2 - Functions 10
3.3 - Services 13
3.4 Classes 14
Task 4 – SQLite and AVD 17
4.1 - SQLite 17
4.2 - AVD 18
Task 5 – Application 19

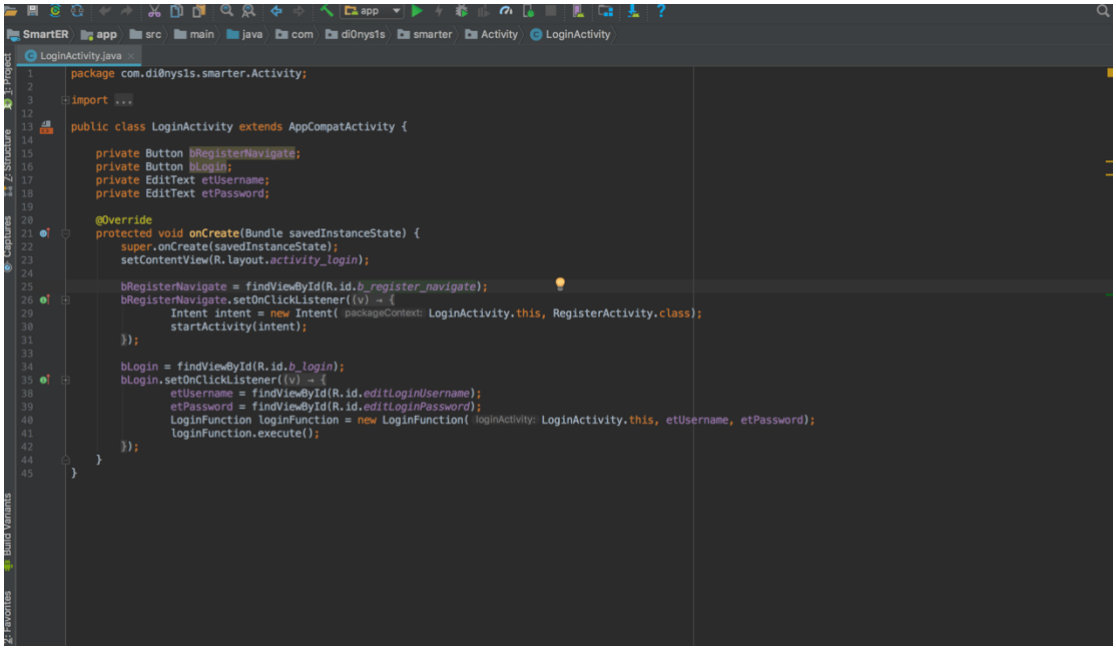
Task 1 – Activity

1. Main Activity



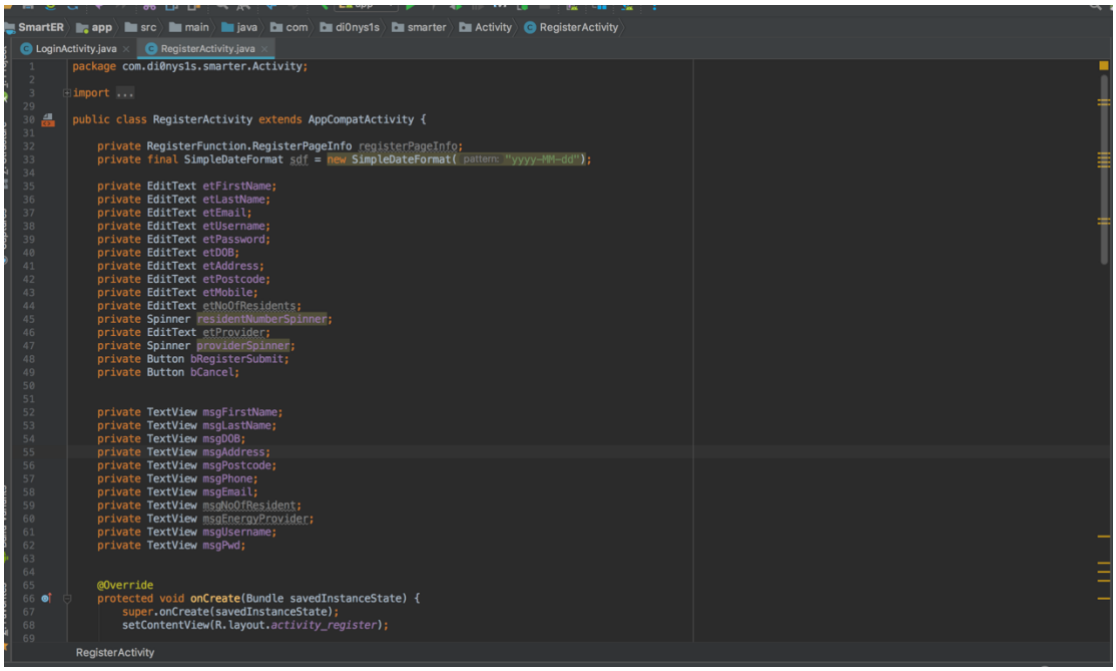
```
package com.di0nys1s.smarter.Activity;

import ...

public class MainActivity extends AppCompatActivity
        implements NavigationView.OnNavigationItemSelectedListener {

    //static boolean isLogin = false;
    private View view;

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

        Toolbar toolbar = findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        DrawerLayout drawer = findViewById(R.id.drawer_layout);
        ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
                activity: this, drawer, toolbar, "Open", "Close");
        drawer.addDrawerListener(toggle);
        toggle.syncState();

        NavigationView navigationView = (NavigationView) findViewById(R.id.nav_view);
        navigationView.setNavigationItemSelectedListener(this);

        getSupportActionBar().setTitle("SmartER");
        FragmentManager fragmentManager = getFragmentManager();
        fragmentManager.beginTransaction().replace(R.id.content_frame, new MainFragment()).commit();

    }

    @Override
    public void onBackPressed() {...}

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {...}

    @SuppressWarnings("StatementWithEmptyBody")
    @Override
    public boolean onNavigationItemSelected(MenuItem item) {...}
}
```

2. Login Activity

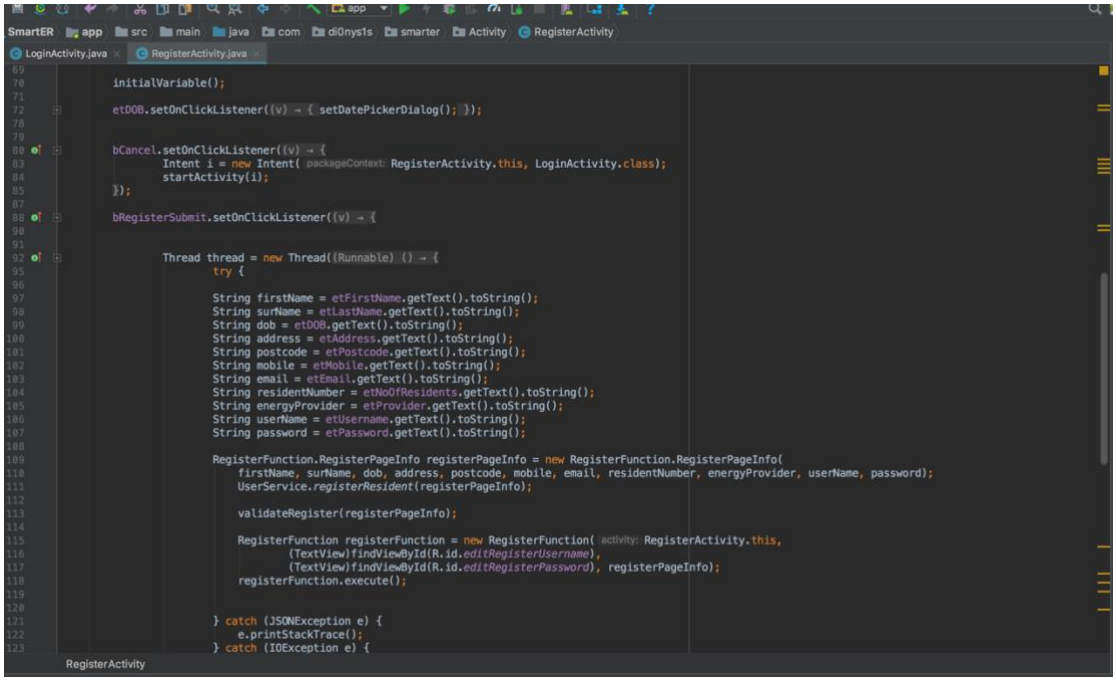


```
package com.di0nys1s.smarter.Activity;

import ...

public class LoginActivity extends AppCompatActivity {

    private Button bRegisterNavigate;
    private Button bLogin;
    private EditText etUsername;
    private EditText etPassword;

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

        bRegisterNavigate = findViewById(R.id.b_register_navigate);
        bRegisterNavigate.setOnClickListener((v) -> {
                Intent intent = new Intent( packageContext: LoginActivity.this, RegisterActivity.class);
                startActivity(intent);
        });

        bLogin = findViewById(R.id.b_login);
        bLogin.setOnClickListener((v) -> {
                etUsername = findViewById(R.id.editLoginUsername);
                etPassword = findViewById(R.id.editLoginPassword);
                LoginFunction loginFunction = new LoginFunction( loginActivity: LoginActivity.this, etUsername, etPassword);
                loginFunction.execute();
        });
    }
}
```

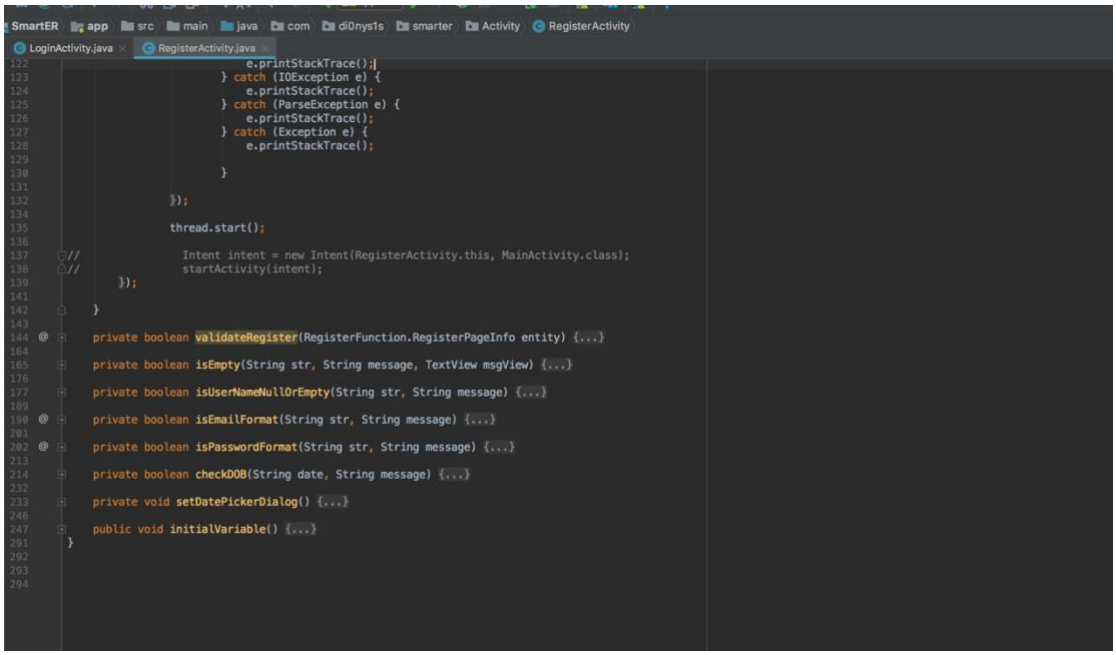
3. Register Activity



```
package com.di0nys1s.smarter.Activity;

import ...

public class RegisterActivity extends AppCompatActivity {

    private RegisterFunction.RegisterPageInfo registerPageInfo;
    private final SimpleDateFormat sdf = new SimpleDateFormat( pattern: "yyyy-MM-dd");

    private EditText etFirstName;
    private EditText etLastName;
    private EditText etEmail;
    private EditText etUsername;
    private EditText etPassword;
    private EditText etDOB;
    private EditText etAddress;
    private EditText etPostcode;
    private EditText etMobile;
    private EditText etNoOfResidents;
    private Spinner residentNumberSpinner;
    private EditText etProvider;
    private Spinner providerSpinner;
    private Button bRegisterSubmit;
    private Button bCancel;


    private TextView msgFirstName;
    private TextView msgLastName;
    private TextView msgDOB;
    private TextView msgAddress;
    private TextView msgPostcode;
    private TextView msgPhone;
    private TextView msgEmail;
    private TextView msgNoOfResident;
    private TextView msgEnergyProvider;
    private TextView msgUsername;
    private TextView msgPwd;


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

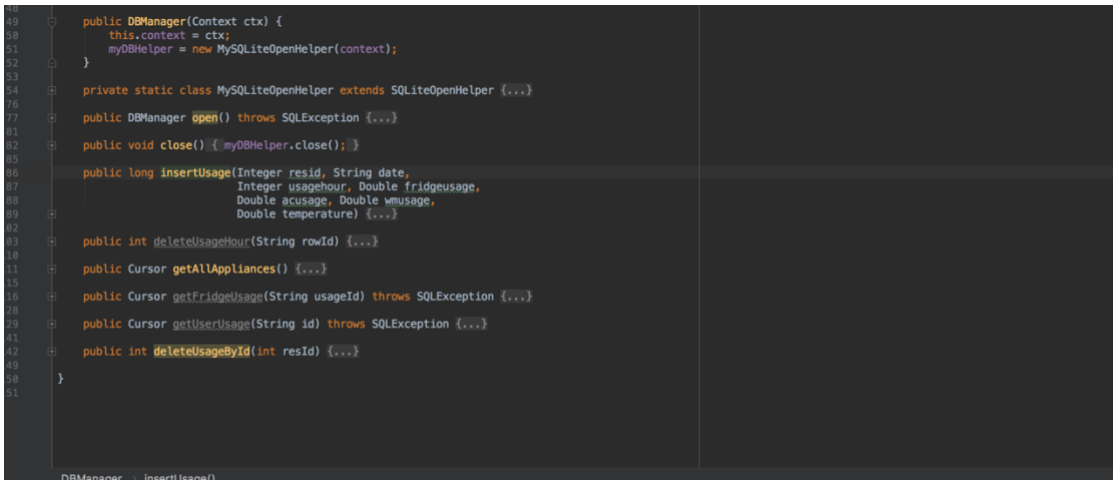


```
        initialVariable();

        etDOB.setOnClickListener((v) -> { setDatePickerDialog(); });


        bCancel.setOnClickListener((v) -> {
                Intent i = new Intent( packageContext: RegisterActivity.this, LoginActivity.class);
                startActivity(i);
        });

        bRegisterSubmit.setOnClickListener((v) -> {


                Thread thread = new Thread((Runnable) () -> {
                        try {

                        String firstName = etFirstName.getText().toString();
                        String surName = etLastName.getText().toString();
                        String dob = etDOB.getText().toString();
                        String address = etAddress.getText().toString();
                        String postcode = etPostcode.getText().toString();
                        String mobile = etMobile.getText().toString();
                        String email = etEmail.getText().toString();
                        String residentNumber = etNoOfResidents.getText().toString();
                        String energyProvider = etProvider.getText().toString();
                        String userName = etUsername.getText().toString();
                        String password = etPassword.getText().toString();

                        RegisterFunction.RegisterPageInfo registerPageInfo = new RegisterFunction.RegisterPageInfo(
                            firstName, surName, dob, address, postcode, mobile, email, residentNumber, energyProvider, userName, password);
                            UserService.registerResident(registerPageInfo);

                            validateRegister(registerPageInfo);

                            RegisterFunction registerFunction = new RegisterFunction( activity: RegisterActivity.this,
                                    (TextView)findViewById(R.id.editRegisterUsername),
                                    (TextView)findViewById(R.id.editRegisterPassword), registerPageInfo);
                            registerFunction.execute();


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

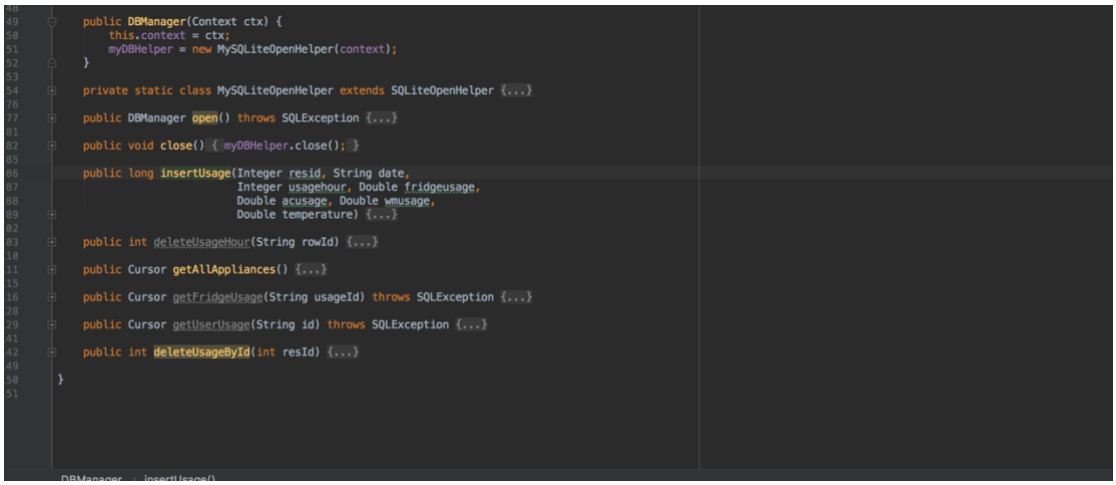


Task 2 – DB Manager and DB Structure

DB MANAGER



DB STRUCTURE

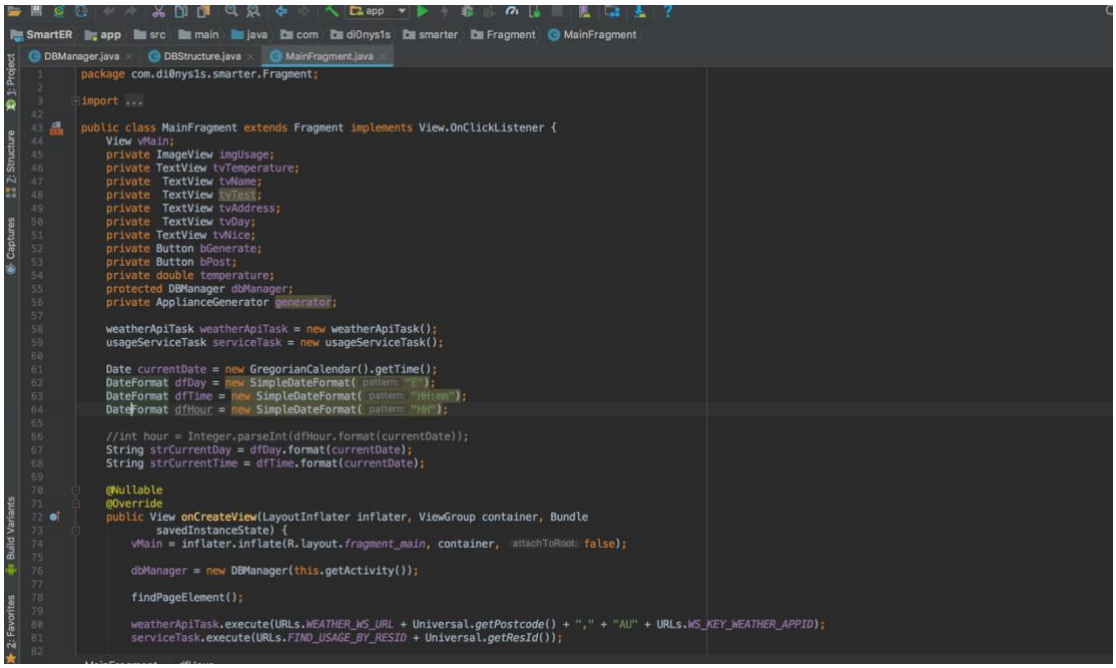


```
    public DBManager(Context ctx) {
        this.context = ctx;
        myDBHelper = new MySQLiteOpenHelper(context);
    }

    private static class MySQLiteOpenHelper extends SQLiteOpenHelper {...}

    public DBManager open() throws SQLException {...}

    public void close() { myDBHelper.close(); }

    public long insertUsage(Integer resid, String date,
                            Integer usagehour, Double fridgeusage,
                            Double acusage, Double wmusage,
                            Double temperature) {...}

    public int deleteUsageHour(String rowId) {...}

    public Cursor getAllAppliances() {...}

    public Cursor getFridgeUsage(String usageId) throws SQLException {...}

    public Cursor getUserUsage(String id) throws SQLException {...}

    public int deleteUsageById(int resId) {...}

}
```

Task 3 – Fragments, Functions, Services and Classes

3.1 Fragments

3.1.1 Main Fragment

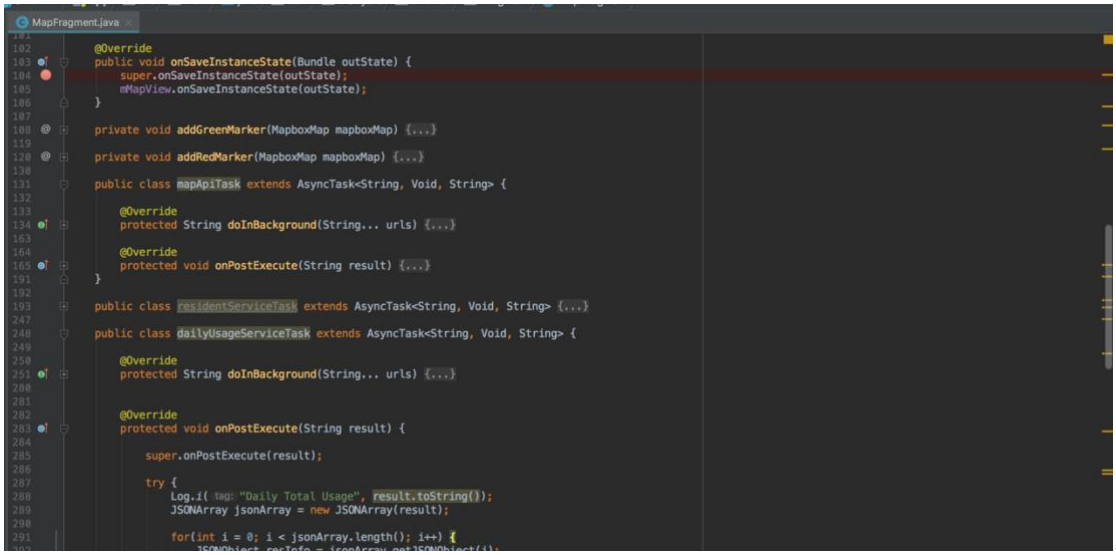


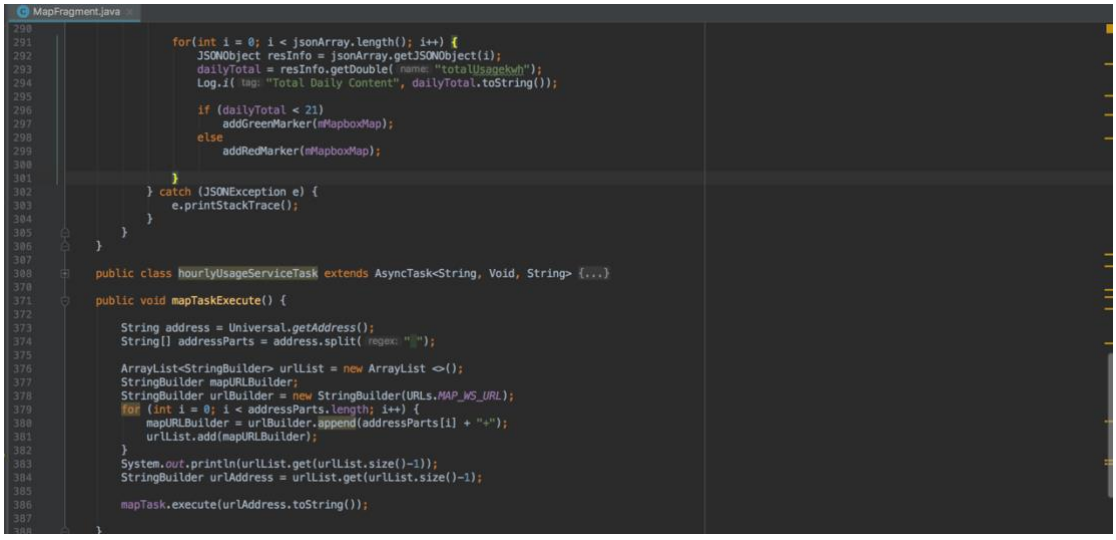
```
package com.di0nys1s.smarter.Fragment;

import ...

public class MainFragment extends Fragment implements View.OnClickListener {
    View vMain;
    private ImageView imgUsage;
    private TextView tvTemperature;
    private  TextView tvName;
    private  TextView tvTest;
    private  TextView tvAddress;
    private  TextView tvDay;
    private TextView tvNice;
    private Button bGenerate;
    private Button bPost;
    private double temperature;
    protected DBManager dbManager;
    private ApplianceGenerator generator;

    weatherApiTask weatherApiTask = new weatherApiTask();
    usageServiceTask serviceTask = new usageServiceTask();

    Date currentDate = new GregorianCalendar().getTime();
    DateFormat dfDay = new SimpleDateFormat( pattern: "E");
    DateFormat dfTime = new SimpleDateFormat( pattern: "HH:mm");
    DateFormat dfHour = new SimpleDateFormat( pattern: "HH");

    //int hour = Integer.parseInt(dfHour.format(currentDate));
    String strCurrentDay = dfDay.format(currentDate);
    String strCurrentTime = dfTime.format(currentDate);

    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle
            savedInstanceState) {
        vMain = inflater.inflate(R.layout.fragment_main, container,  attachToRoot: false);

        dbManager = new DBManager(this.getActivity());

        findPageElement();

        weatherApiTask.execute(URLs.WEATHER_WS_URL + Universal.getPostcode() + "," + "AU" + URLs.WS_KEY_WEATHER_APPID);
        serviceTask.execute(URLs.FIND_USAGE_BY_RESID + Universal.getResId());
```

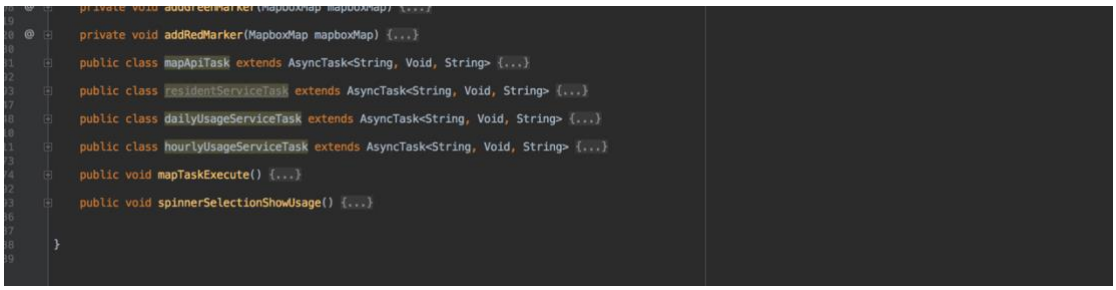
```
DBManager.java  DBStructure.java  MainFragment.java
80          weatherApiTask.execute(URLs.WEATHER_WS_URL + Universal.getPostcode() + "," + "AU" + URLs.WS_KEY_WEATHER_APPID);
81          serviceTask.execute(URLs.FIND_USAGE_BY_RESID + Universal.getResId());
82
83          pickCurrentDay();
84          showUserInformation();
85
86          bGenerate.setOnClickListener(this);
87          bPost.setOnClickListener(this);
88
89          return vMain;
90      }
91
92      @Override
93      public void onClick(View v) {...}
109
110     public class usageServiceTask extends AsyncTask<String, Void, String> {...}
170
171     public class weatherApiTask extends AsyncTask<String, Void, String> {...}
235
236     public void insertApplianceUsageToLocalDB() {...}
266
267     public void postApplianceUsagetoREST() {...}
311
312     public Cursor readData() {...}
326
327     public void deleteDataById(Integer resid) {...}
339
340     private void mainUsageMessage(Double hourlyUsage) {...}
358
359     public void pickCurrentDay() {...}
375
376     public void showUserInformation() {...}
382
383     public void findPageElement() {...}
395
396     public static void createUsage(ElectricityUsage usage){...}
427
428 }
```

3.1.2 Map Fragment

```
MapFragment.java
1   package com.di0nys1s.smarter.Fragment;
2
3   import ...
42
43  public class MapFragment extends Fragment {
44      View vMapFragment;
45      private Double dailyTotal;
46      private Double hourlyTotal;
47      private MapboxMap mMapboxMap;
48      private MapView mMapView;
49      private LatLng userAddressLatLNG;
50
51      mapApiTask mapTask = new mapApiTask();
52      //residentServiceTask residentServiceTask = new residentServiceTask();
53
54      private Spinner spinnerDailyHourly;
55      String types[] = {"daily", "hourly"};
56      ArrayAdapter<String> adapter;
57
58      Date currentDate = new GregorianCalendar().getTime();
59      DateFormat dfTime = new SimpleDateFormat( pattern: "H");
60      String time = dfTime.format(currentDate.getTime());
61      int t = Integer.parseInt(time);
62
63      @{...}
65      public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle
66              savedInstanceState) {...}
83
84      @Override
85      public void onResume() {
86          super.onResume();
87          mMapView.onResume();
88      }
89
90      @Override
91      public void onPause() {
92          super.onPause();
93          mMapView.onPause();
```


MapFragment.java
@Override
public void onSaveInstanceState(Bundle outState) {
super.onSaveInstanceState(outState);
mMapView.onSaveInstanceState(outState);
}
private void addGreenMarker(MapboxMap mapboxMap) {...}
private void addRedMarker(MapboxMap mapboxMap) {...}
public class mapApiTask extends AsyncTask<String, Void, String> {
@Override
protected String doInBackground(String... urls) {...}
@Override
protected void onPostExecute(String result) {...}
}
public class residentServiceTask extends AsyncTask<String, Void, String> {...}
public class dailyUsageServiceTask extends AsyncTask<String, Void, String> {
@Override
protected String doInBackground(String... urls) {...}
@Override
protected void onPostExecute(String result) {
super.onPostExecute(result);
try {
Log.i(tag: "Daily Total Usage", result.toString());
JSONArray jsonArray = new JSONArray(result);
for(int i = 0; i < jsonArray.length(); i++) {


MapFragment.java
for(int i = 0; i < jsonArray.length(); i++) {
JSONObject resInfo = jsonArray.getJSONObject(i);
dailyTotal = resInfo.getDouble(name: "totalUsagekwh");
Log.i(tag: "Total Daily Content", dailyTotal.toString());
if (dailyTotal < 21)
addGreenMarker(mMapboxMap);
else
addRedMarker(mMapboxMap);
}
} catch (JSONException e) {
e.printStackTrace();
}
}
}
public class hourlyUsageServiceTask extends AsyncTask<String, Void, String> {...}
public void mapTaskExecute() {
String address = Universal.getAddress();
String[] addressParts = address.split(regex: " ");
ArrayList<StringBuilder> urlList = new ArrayList <>();
StringBuilder mapURLBuilder;
StringBuilder urlBuilder = new StringBuilder(URLs.MAP_WS_URL);
for (int i = 0; i < addressParts.length; i++) {
mapURLBuilder = urlBuilder.append(addressParts[i] + "+");
urlList.add(mapURLBuilder);
}
System.out.println(urlList.get(urlList.size()-1));
StringBuilder urlAddress = urlList.get(urlList.size()-1);
mapTask.execute(urlAddress.toString());
}


private void addRedMarker(MapboxMap mapboxMap) {...}
public class mapApiTask extends AsyncTask<String, Void, String> {...}
public class residentServiceTask extends AsyncTask<String, Void, String> {...}
public class dailyUsageServiceTask extends AsyncTask<String, Void, String> {...}
public class hourlyUsageServiceTask extends AsyncTask<String, Void, String> {...}
public void mapTaskExecute() {...}
public void spinnerSelectionShowUsage() {...}
}

3.1.3 Report Fragment

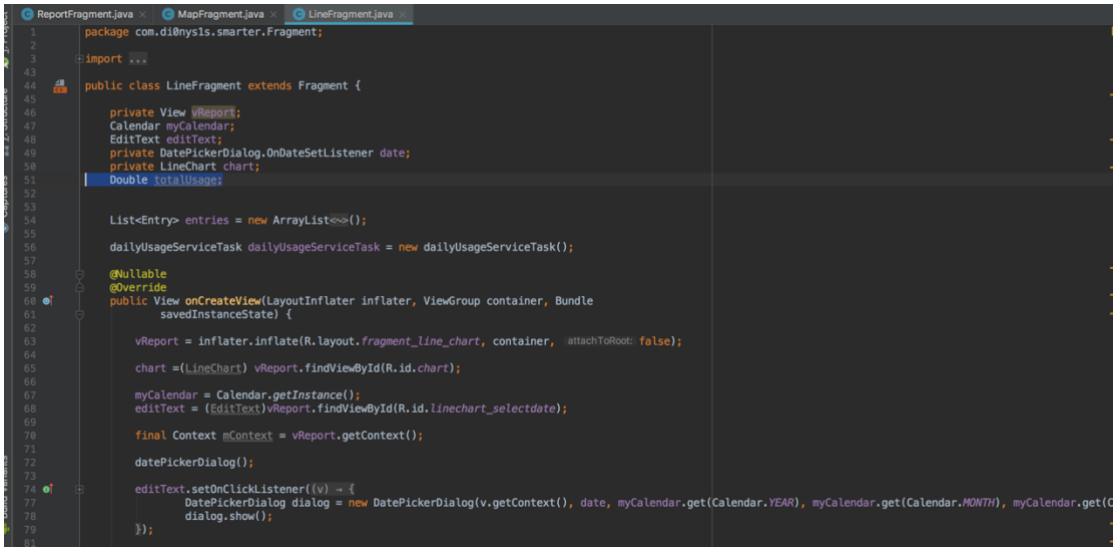
PIE FRAGMENT

ReportFragment.java

```
package com.di0nys1s.smarter.Fragment;

import ...

public class ReportFragment extends Fragment {
    private View vReport;
    private Button bReport;
    private TextView tvReport;
    private PieChart pieChart;
    private Calendar myCalendar;
    private EditText editText;
    private DatePickerDialog.OnDateSetListener date;

    dailyUsageServiceTask dailyUsageServiceTask = new dailyUsageServiceTask();


    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle
            savedInstanceState) {

        vReport = inflater.inflate(R.layout.fragment_report, container,  attachToRoot: false);
        tvReport = vReport.findViewById(R.id.tv_report);

        pieChart =(PieChart) vReport.findViewById(R.id.piechart);
        myCalendar = Calendar.getInstance();
        editText = (EditText)vReport.findViewById(R.id.piechart_selectdate);

        final Context mContext = vReport.getContext();

        dailyUsageServiceTask.execute(URLs.FIND_APP_USAGE_BY_RESID_DATE_URL + Universal.getResId() + "/" + "2018-03-18");

        datePickerDialog();

        editText.setOnClickListener((v) → {
                DatePickerDialog dialog = new DatePickerDialog(v.getContext(), date, myCalendar.get(Calendar.YEAR), myCalendar.get(Calendar.MONTH), myCalendar.get(Cal
                dialog.show();
        });

        editText.addTextChangedListener(new TextWatcher() {...});

        return vReport;
    }
```

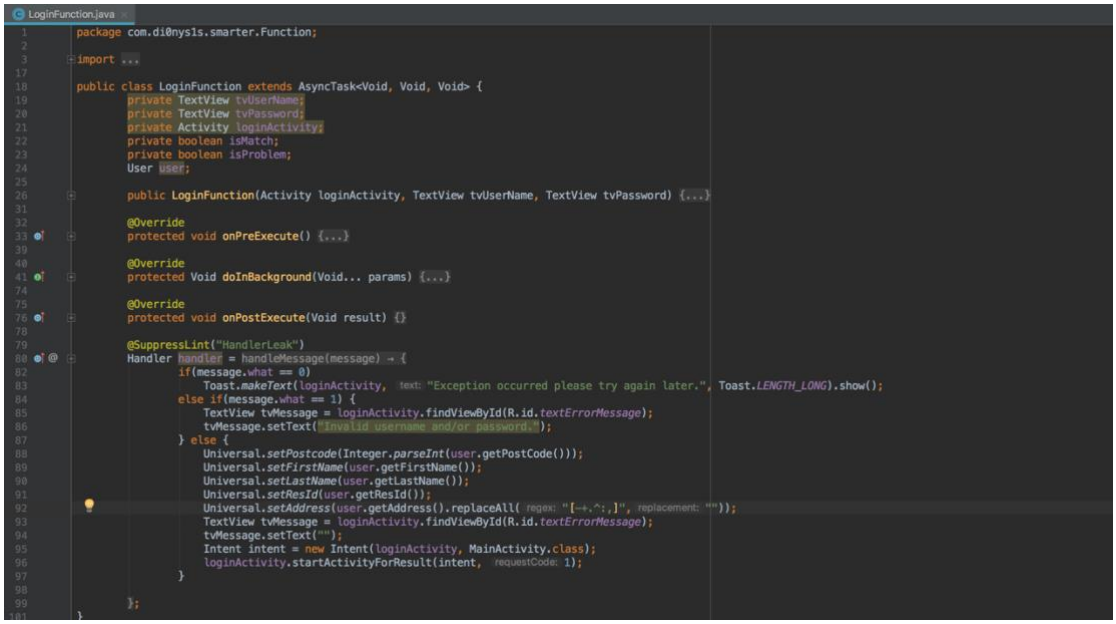
```
    public void datePickerDialog() {...}

    public class dailyUsageServiceTask extends AsyncTask<String, Void, String> {...}

}
```

LINE FRAGMENT



3.2 - Functions

LOGIN FUNCTION



REGISTER FUNCTION

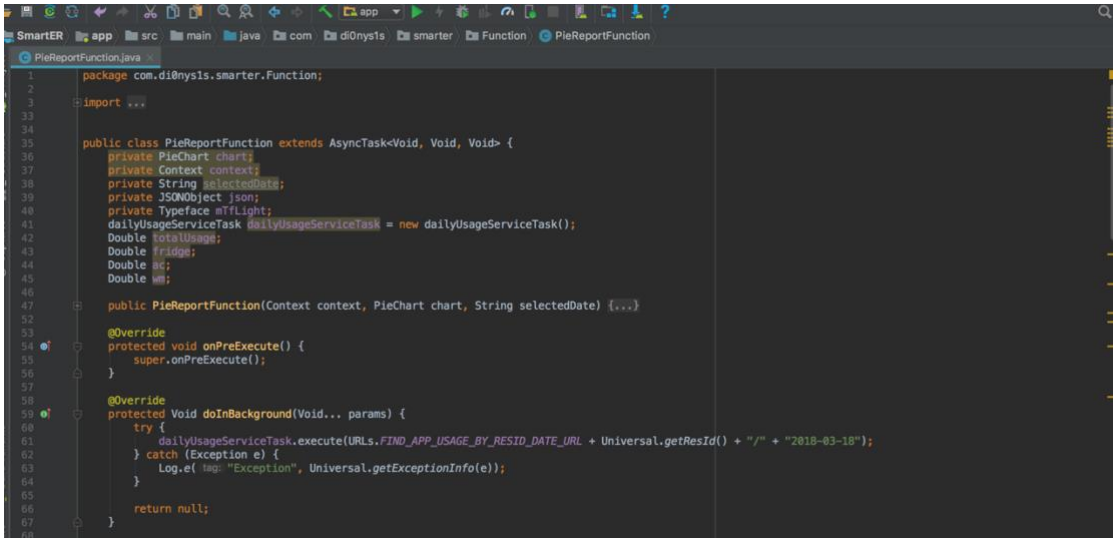
RegisterFunction.java

```
import ...

public class RegisterFunction  extends AsyncTask<Void, Void, Void> {
    private String username;
    private String email;
    private TextView tvUserName;
    private TextView tvEmail;
    private Activity activity;

    private RegisterFunction.RegisterPageInfo registerPageInfo;
    boolean isSuccess;

    public RegisterFunction(Activity activity, TextView tvUserName, TextView tvEmail, RegisterFunction.RegisterPageInfo registerPageInfo) {...}

    protected Void doInBackground(Void... params) {...}

    @Override
    protected void onPostExecute(Void result) {}
    @SuppressLint("HandlerLeak")
    Handler handler = handleMessage(message) → {
            if(message.what == 0) {
                try {
                    if (isSuccess) {
                        Universal.setAddress(registerPageInfo.getAddress());
                        Universal.setFirstName(registerPageInfo.getFirstName());
                        Intent intent = new Intent(activity, MainActivity.class);
                        activity.startActivityForResult(intent,  requestCode: 1);
                    } else {
                        Toast.makeText(activity,  text: "Error occurs in the page. Please try again.", Toast.LENGTH_LONG);
                    }
                } catch (Exception ex) {
                    Toast.makeText(activity,  text: "Error occurs in the page. Please try again.", Toast.LENGTH_LONG);
                }
            } else if(message.what == 1) {
                tvEmail.setBackgroundColor(activity.getResources().getColor(R.color.errorBackground));
                ((TextView)activity.findViewById(R.id.textEmailErrorMsg)).setText("User with same email exists");
            } else {
                tvUserName.setBackgroundColor(activity.getResources().getColor(R.color.errorBackground));
                ((TextView)activity.findViewById(R.id.textUserNameErrorMsg)).setText("User name exists");
            }

    };

    public static class RegisterPageInfo {...}
```

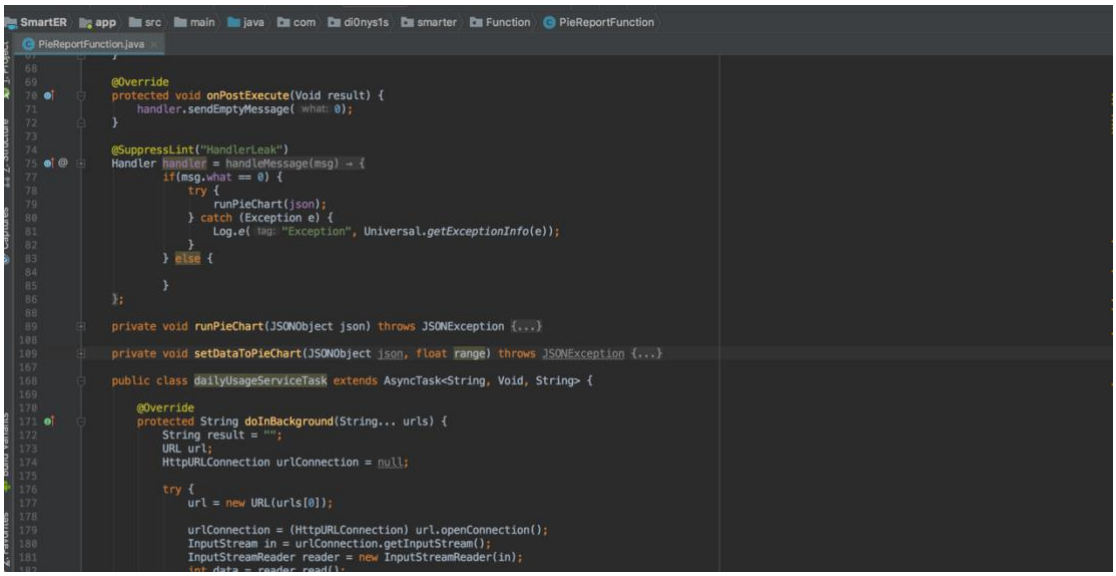
RegisterFunction.java

```
import ...

public class RegisterFunction  extends AsyncTask<Void, Void, Void> {
    private String username;
    private String email;
    private TextView tvUserName;
    private TextView tvEmail;
    private Activity activity;

    private RegisterFunction.RegisterPageInfo registerPageInfo;
    boolean isSuccess;

    public RegisterFunction(Activity activity, TextView tvUserName, TextView tvEmail, RegisterFunction.RegisterPageInfo registerPageInfo) {...}

    protected Void doInBackground(Void... params) {...}

    @Override
    protected void onPostExecute(Void result) {}
    @SuppressLint("HandlerLeak")
    Handler handler = handleMessage(message) → {
            if(message.what == 0) {
                try {
                    if (isSuccess) {
                        Universal.setAddress(registerPageInfo.getAddress());
                        Universal.setFirstName(registerPageInfo.getFirstName());
                        Intent intent = new Intent(activity, MainActivity.class);
                        activity.startActivityForResult(intent,  requestCode: 1);
                    } else {
                        Toast.makeText(activity,  text: "Error occurs in the page. Please try again.", Toast.LENGTH_LONG);
                    }
                } catch (Exception ex) {
                    Toast.makeText(activity,  text: "Error occurs in the page. Please try again.", Toast.LENGTH_LONG);
                }
            } else if(message.what == 1) {
                tvEmail.setBackgroundColor(activity.getResources().getColor(R.color.errorBackground));
                ((TextView)activity.findViewById(R.id.textEmailErrorMsg)).setText("User with same email exists");
            } else {
                tvUserName.setBackgroundColor(activity.getResources().getColor(R.color.errorBackground));
                ((TextView)activity.findViewById(R.id.textUserNameErrorMsg)).setText("User name exists");
            }

    };

    public static class RegisterPageInfo {...}
```

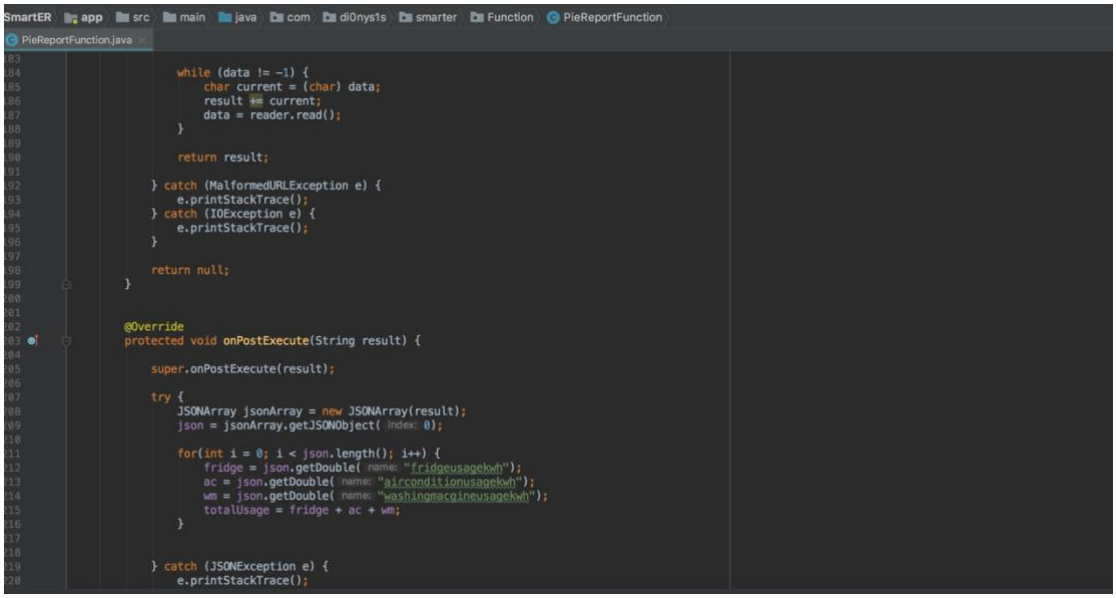
PIE REPORT FUNCTION



```
package com.di0nys1s.smarter.Function;

import ...


public class PieReportFunction extends AsyncTask<Void, Void, Void> {
    private PieChart chart;
    private Context context;
    private String selectedDate;
    private JSONObject json;
    private Typeface mTfLight;
    dailyUsageServiceTask dailyUsageServiceTask = new dailyUsageServiceTask();
    Double totalUsage;
    Double fridge;
    Double ac;
    Double wm;

    public PieReportFunction(Context context, PieChart chart, String selectedDate) {...}

    @Override
    protected void onPreExecute() {
        super.onPreExecute();
    }

    @Override
    protected Void doInBackground(Void... params) {
        try {
            dailyUsageServiceTask.execute(URLs.FIND_APP_USAGE_BY_RESID_DATE_URL + Universal.getResId() + "/" + "2018-03-18");
        } catch (Exception e) {
            Log.e( tag: "Exception", Universal.getExceptionInfo(e));
        }

        return null;
    }
```



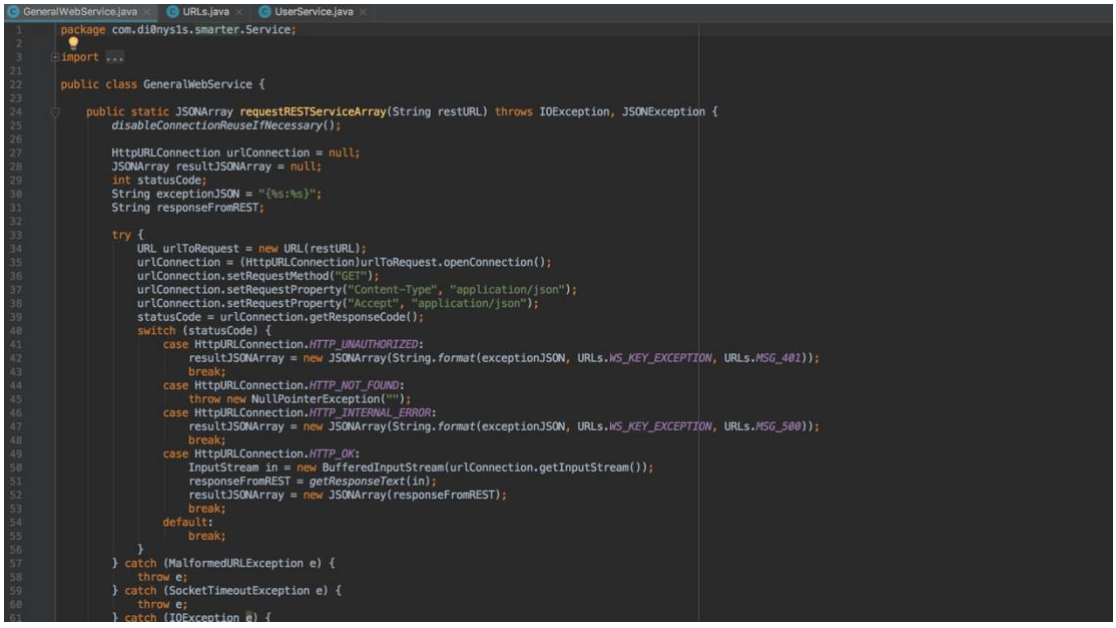
```
    @Override
    protected void onPostExecute(Void result) {
        handler.sendEmptyMessage( what: 0);
    }

    @SuppressLint("HandlerLeak")
    Handler handler = handleMessage(msg) -> {
            if(msg.what == 0) {
                try {
                    runPieChart(json);
                } catch (Exception e) {
                    Log.e( tag: "Exception", Universal.getExceptionInfo(e));
                }
            } else {

            }
    };

    private void runPieChart(JSONObject json) throws JSONException {...}

    private void setDataToPieChart(JSONObject json, float range) throws JSONException {...}

    public class dailyUsageServiceTask extends AsyncTask<String, Void, String> {

        @Override
        protected String doInBackground(String... urls) {
            String result = "";
            URL url;
            HttpURLConnection urlConnection = null;

            try {
                url = new URL(urls[0]);

                urlConnection = (HttpURLConnection) url.openConnection();
                InputStream in = urlConnection.getInputStream();
                InputStreamReader reader = new InputStreamReader(in);
```



```
                while (data != -1) {
                    char current = (char) data;
                    result += current;
                    data = reader.read();
                }

                return result;

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

            return null;
        }


        @Override
        protected void onPostExecute(String result) {

            super.onPostExecute(result);

            try {
                JSONArray jsonArray = new JSONArray(result);
                json = jsonArray.getJSONObject( index: 0);

                for(int i = 0; i < json.length(); i++) {
                    fridge = json.getDouble( name: "fridgeusagekwh");
                    ac = json.getDouble( name: "airconditionusagekwh");
                    wm = json.getDouble( name: "washingmacgineusagekwh");
                    totalUsage = fridge + ac + wm;
                }


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

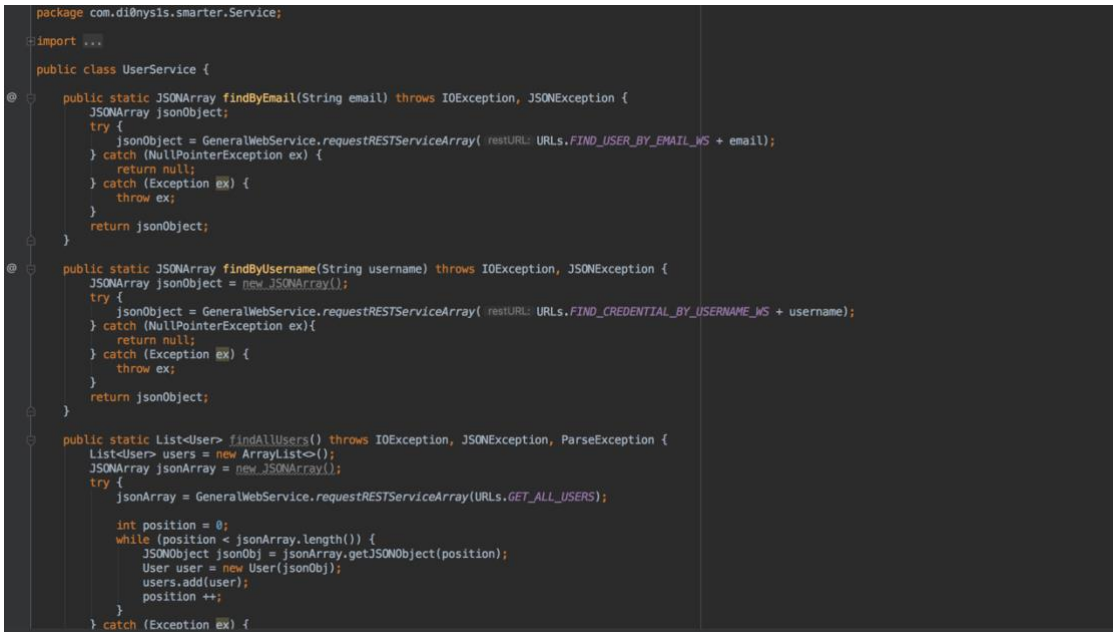
3.3 - Services

GENERAL WEB SERVICE



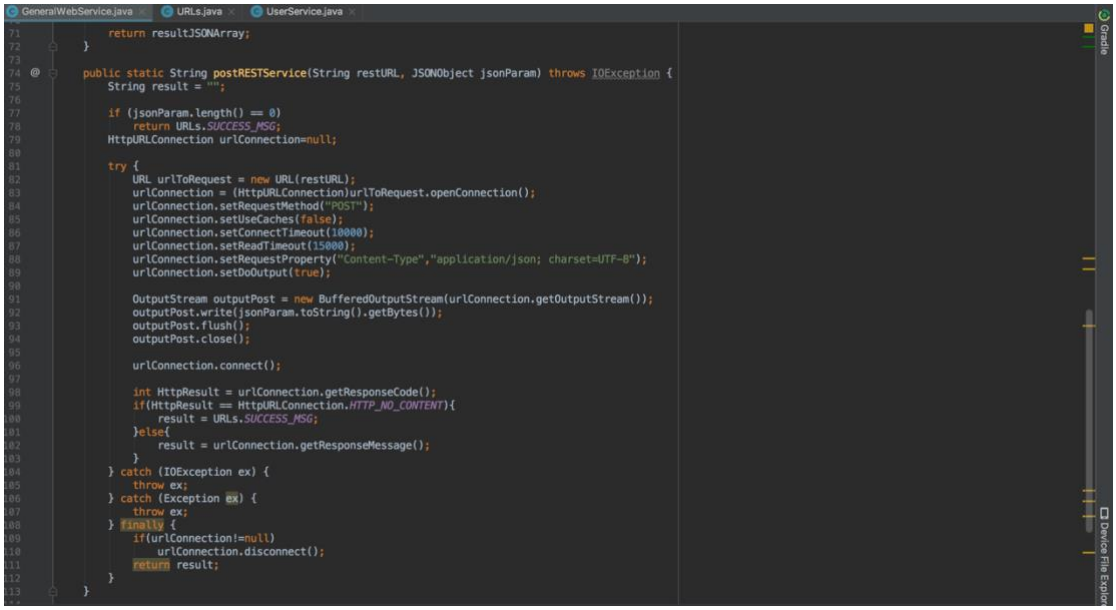
```java
package com.di0nys1s.smarter.Service;

import ...

public class GeneralWebService {

    public static JSONArray requestRESTServiceArray(String restURL) throws IOException, JSONException {
        disableConnectionReuseIfNecessary();

        HttpURLConnection urlConnection = null;
        JSONArray resultJSONArray = null;
        int statusCode;
        String exceptionJSON = "{%s:%s}";
        String responseFromREST;

        try {
            URL urlToRequest = new URL(restURL);
            urlConnection = (HttpURLConnection)urlToRequest.openConnection();
            urlConnection.setRequestMethod("GET");
            urlConnection.setRequestProperty("Content-Type", "application/json");
            urlConnection.setRequestProperty("Accept", "application/json");
            statusCode = urlConnection.getResponseCode();
            switch (statusCode) {
                case HttpURLConnection.HTTP_UNAUTHORIZED:
                    resultJSONArray = new JSONArray(String.format(exceptionJSON, URLs.WS_KEY_EXCEPTION, URLs.MSG_401));
                    break;
                case HttpURLConnection.HTTP_NOT_FOUND:
                    throw new NullPointerException("");
                case HttpURLConnection.HTTP_INTERNAL_ERROR:
                    resultJSONArray = new JSONArray(String.format(exceptionJSON, URLs.WS_KEY_EXCEPTION, URLs.MSG_500));
                    break;
                case HttpURLConnection.HTTP_OK:
                    InputStream in = new BufferedInputStream(urlConnection.getInputStream());
                    responseFromREST = getResponseText(in);
                    resultJSONArray = new JSONArray(responseFromREST);
                    break;
                default:
                    break;
            }
        } catch (MalformedURLException e) {
            throw e;
        } catch (SocketTimeoutException e) {
            throw e;
        } catch (IOException e) {
```

USER SERVICE



```java
package com.di0nys1s.smarter.Service;

import ...

public class UserService {

    public static JSONArray findByEmail(String email) throws IOException, JSONException {
        JSONArray jsonObject;
        try {
            jsonObject = GeneralWebService.requestRESTServiceArray( restURL: URLs.FIND_USER_BY_EMAIL_WS + email);
        } catch (NullPointerException ex) {
            return null;
        } catch (Exception ex) {
            throw ex;
        }
        return jsonObject;
    }

    public static JSONArray findByUsername(String username) throws IOException, JSONException {
        JSONArray jsonObject = new JSONArray();
        try {
            jsonObject = GeneralWebService.requestRESTServiceArray( restURL: URLs.FIND_CREDENTIAL_BY_USERNAME_WS + username);
        } catch (NullPointerException ex){
            return null;
        } catch (Exception ex) {
            throw ex;
        }
        return jsonObject;
    }

    public static List<User> findAllUsers() throws IOException, JSONException, ParseException {
        List<User> users = new ArrayList<>();
        JSONArray jsonArray = new JSONArray();
        try {
            jsonArray = GeneralWebService.requestRESTServiceArray(URLs.GET_ALL_USERS);

            int position = 0;
            while (position < jsonArray.length()) {
                JSONObject jsonObj = jsonArray.getJSONObject(position);
                User user = new User(jsonObj);
                users.add(user);
                position ++;
            }
        } catch (Exception ex) {
```

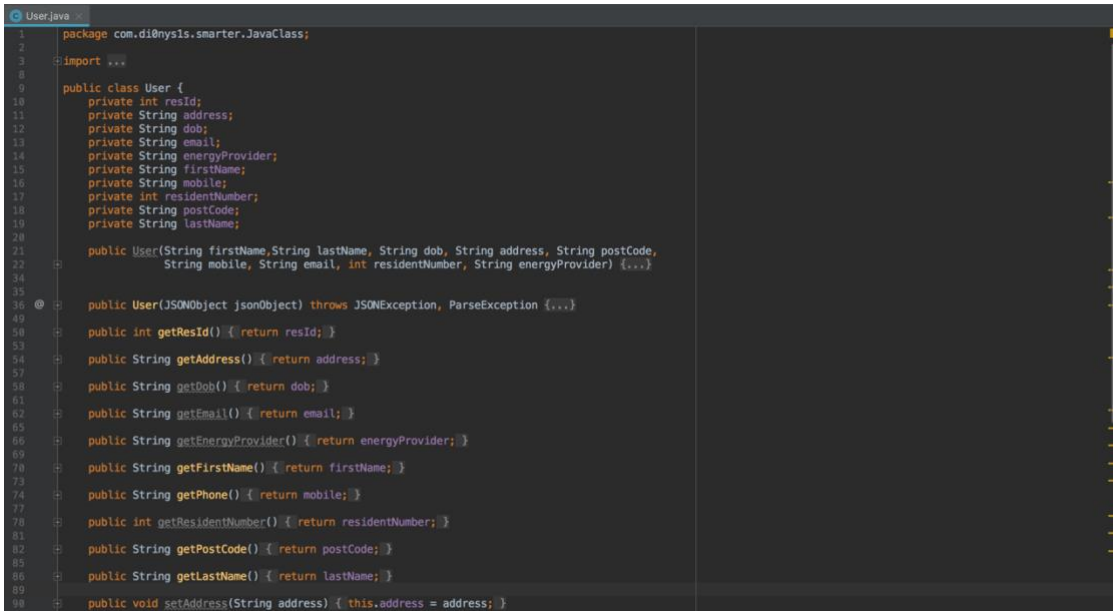
POST REST SERVICE




```
        return resultJSONArray;
    }

    public static String postRESTService(String restURL, JSONObject jsonParam) throws IOException {
        String result = "";

        if (jsonParam.length() == 0)
            return URLs.SUCCESS_MSG;
        HttpURLConnection urlConnection=null;

        try {
            URL urlToRequest = new URL(restURL);
            urlConnection = (HttpURLConnection)urlToRequest.openConnection();
            urlConnection.setRequestMethod("POST");
            urlConnection.setUseCaches(false);
            urlConnection.setConnectTimeout(10000);
            urlConnection.setReadTimeout(15000);
            urlConnection.setRequestProperty("Content-Type","application/json; charset=UTF-8");
            urlConnection.setDoOutput(true);

            OutputStream outputPost = new BufferedOutputStream(urlConnection.getOutputStream());
            outputPost.write(jsonParam.toString().getBytes());
            outputPost.flush();
            outputPost.close();

            urlConnection.connect();

            int HttpResult = urlConnection.getResponseCode();
            if(HttpResult == HttpURLConnection.HTTP_NO_CONTENT){
                result = URLs.SUCCESS_MSG;
            }else{
                result = urlConnection.getResponseMessage();
            }
        } catch (IOException ex) {
            throw ex;
        } catch (Exception ex) {
            throw ex;
        } finally {
            if(urlConnection!=null)
                urlConnection.disconnect();
            return result;
        }
    }
```

3.4 Classes

USER




```
package com.di0nys1s.smarter.JavaClass;

import ...

public class User {
    private int resId;
    private String address;
    private String dob;
    private String email;
    private String energyProvider;
    private String firstName;
    private String mobile;
    private int residentNumber;
    private String postCode;
    private String lastName;

    public User(String firstName,String lastName, String dob, String address, String postCode,
                String mobile, String email, int residentNumber, String energyProvider) {...}


    public User(JSONObject jsonObject) throws JSONException, ParseException {...}

    public int getResId() { return resId; }

    public String getAddress() { return address; }

    public String getDob() { return dob; }

    public String getEmail() { return email; }

    public String getEnergyProvider() { return energyProvider; }

    public String getFirstName() { return firstName; }

    public String getPhone() { return mobile; }

    public int getResidentNumber() { return residentNumber; }

    public String getPostCode() { return postCode; }

    public String getLastName() { return lastName; }

    public void setAddress(String address) { this.address = address; }
```

```
    public void setDob(String dob) { this.dob = dob; }

    public void setEmail(String email) { this.email = email; }

    public void setEnergyProvider(String energyProvider) { this.energyProvider = energyProvider; }

    public void setFirstName(String firstName) { this.firstName = firstName; }

    public void setPhone(String mobile) { this.mobile = mobile; }

    public void setResidentNumber(int residentNumber) { this.residentNumber = residentNumber; }

    public void setPostCode(String postCode) { this.postCode = postCode; }

    public void setLastName(String lastName) { this.lastName = lastName; }
}
```

UNIVERSAL

Universal.java

```
package com.di0nys1s.smarter.JavaClass;

import ...

public class Universal extends Application {

    private static int resId;
    private static String address;
    private static String firstName;
    private static String lastName;
    private static Context mContext;
    private static int postcode;

    public static String getExceptionInfo(Exception ex) {...}

    public static int getResId() { return resId; }
    public static void setResId(int resId) { Universal.resId = resId; }

    public static String getFirstName() { return firstName; }
    public static void setFirstName(String firstName) { Universal.firstName = firstName; }

    public static String getLastName() { return lastName; }
    public static void setLastName(String lastName) { Universal.lastName = lastName; }

    public static String getAddress() { return address; }
    public static void setAddress(String address) { Universal.address = address; }

    public static Context getmContext() { return mContext; }
    public static void setmContext(Context mContext) { Universal.mContext = mContext; }

    public static int getPostcode() { return postcode; }
    public static void setPostcode(int postcode) { Universal.postcode = postcode; }

    public static boolean isEmptyOrNull (String str) { return str == null || "".equals(str); }

}
```

APPLIANCE GENERATOR

ApplianceGenerator.java

```
package com.di0nys1s.smarter.JavaClass;

import java.util.Random;

public class ApplianceGenerator {
    private double randomFridge;
    private  double randomWM;
    private double randomAC;
    private int randomHour;
    private int hour = 0;

    public double[] getRandomFridgeUsage() {

        double minFridge = 0.3;
        double maxFridge = 0.8;
        double fridge = minFridge + Math.random() * (maxFridge - minFridge);
        randomFridge = Math.round(fridge * 100.00) / 100.0;

        double[] fridgeUsages = new double[24];

        while (hour <= 23) {
            for (int i  = hour; i <= hour; i++) {
                fridgeUsages[i] = randomFridge;
            }
            hour++;
        }

        return fridgeUsages;
    }

    public double[] getRandomWmUsage() {
        double minWM = 0.4;
        double maxWM = 1.3;
        double wm = minWM + Math.random() * (maxWM - minWM);
        randomWM = Math.round(wm * 100.00) / 100.0;
        double zeroWM = 0.0;
        int maxHour = 21;
        int minHour = 6;

        double[] wmUsages = new double[24];

        boolean isHour = true;
        while (isHour) {
            double h = (minHour + (Math.random() * (maxHour - minHour + 1)));
```

ApplianceGenerator.java

```
            double h = (minHour + (Math.random() * (maxHour - minHour + 1)));
            randomHour = (int) (Math.round(h * 100) / 100);
            int hour = 0;
            while (hour <= 23) {
                for (int i  = hour; i <= hour; i++) {
                    wmUsages[i] = zeroWM;
                }
                hour++;
                if (randomHour == hour && randomHour <= 19) {
                    for (int counter = 0; counter <= 2; counter++) {
                        for (int i  = randomHour; i <= 19; i++) {
                            wmUsages[i] = randomWM;
                        }
                        isHour = false;
                        randomHour++;
                        hour++;
                    }
                }
                else if (randomHour == 20) {
                    for (int counter = 0; counter <= 1; counter++) {
                        wmUsages[19] = randomWM;
                        isHour = false;
                        randomHour++;
                        hour++;
                    }
                }
                else if (randomHour == 21) {
                    wmUsages[20] = randomWM;
                    isHour = false;
                    randomHour++;
                    hour++;
                }
            }
        }

        return wmUsages;
    }

    public double[] getRandomAcUsage(Double temperature) {
        double minAC = 1;
        double maxAC = 5;
        double ac = minAC + Math.random() * (maxAC - minAC);
        randomAC = Math.round(ac * 100.00) / 100.0;
        double zeroAC = 0.0;
```

ApplianceGenerator.java

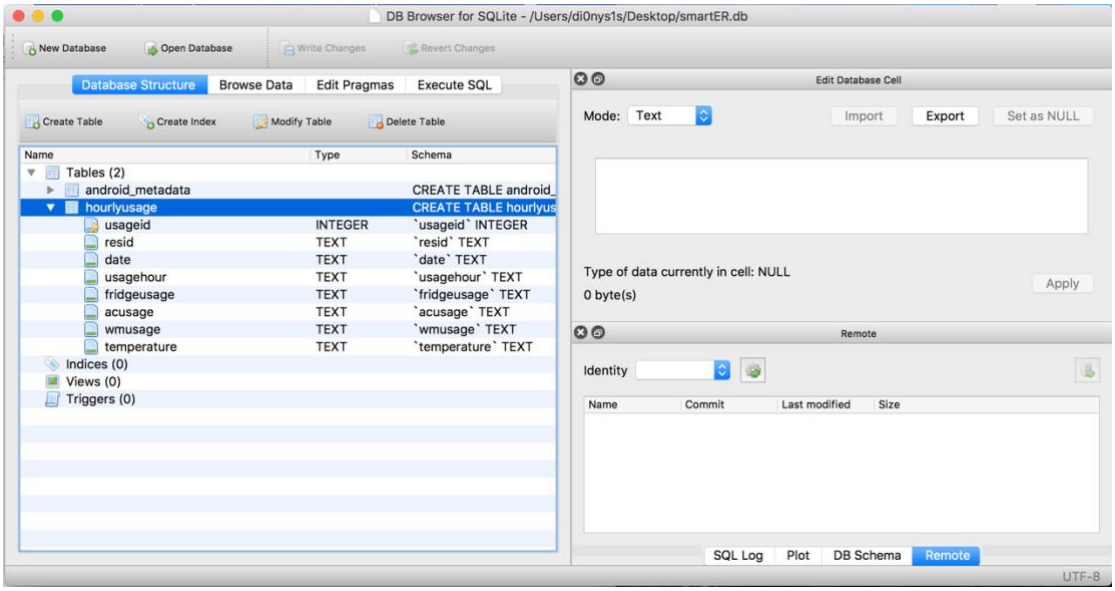
```
        int maxHourAC = 23;
        int minHourAC = 9;
        Random random = new Random();
        Boolean isAcOpen;
        int counter = 0;

        double[] acUsages = new double[24];

        while (hour <= maxHourAC) {
            if (temperature < 20) {
                while (hour <= maxHourAC) {
                    for (int i  = hour; i <= maxHourAC; i++) {
                        acUsages[i] = zeroAC;
                    }
                    hour++;
                }
            } else {
                if (hour < minHourAC) {
                    for (int i  = hour; i < minHourAC; i++) {
                        acUsages[i] = zeroAC;
                    }
                    hour++;
                }
                if (hour >= minHourAC) {
                    isAcOpen = random.nextBoolean();
                    while (hour <= maxHourAC) {
                        while(counter < 10 && hour <= 23) {
                            if (isAcOpen) {
                                for (int i  = hour; i <= maxHourAC; i++) {
                                    acUsages[i] = randomAC;
                                }
                                hour++;
                                counter++;
                                isAcOpen = random.nextBoolean();
                                continue;
                            } else if (!isAcOpen) {
                                for (int i  = hour; i <= maxHourAC; i++) {
                                    acUsages[i] = zeroAC;
                                }
                                hour++;
                                isAcOpen = random.nextBoolean();
                            }
                        }
```

```
                        }
                        if (hour <= maxHourAC) {
                            while (hour <= maxHourAC) {
                                for (int i  = hour; i <= maxHourAC; i++) {
                                    acUsages[i] = zeroAC;
                                }
                                hour++;
                            }
                        }
                    }

                }
            }
        }
        return acUsages;
    }
}
```

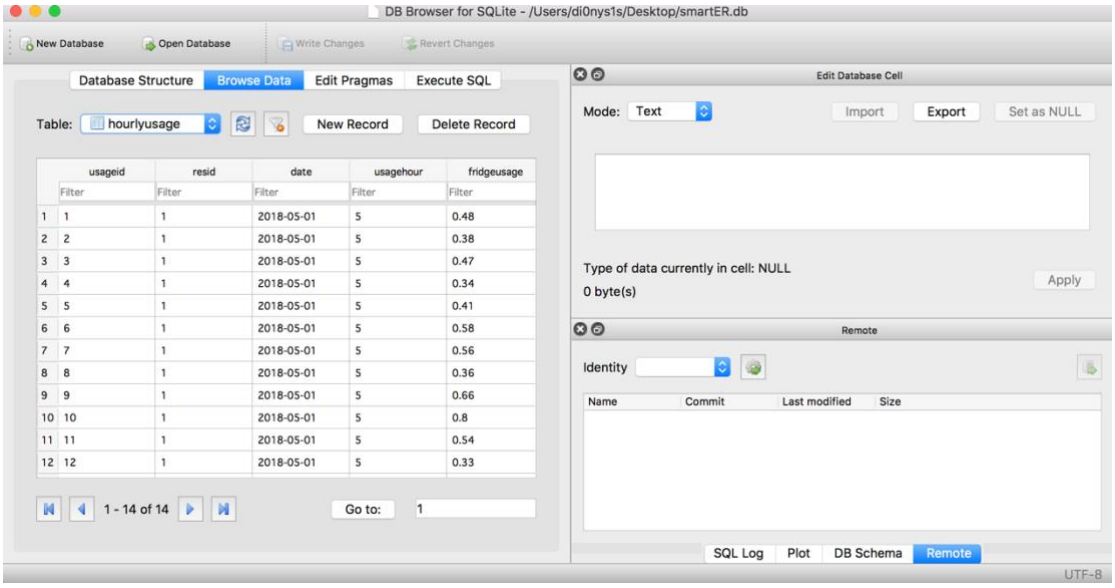
Task 4 – SQLite and AVD

4.1 - SQLite

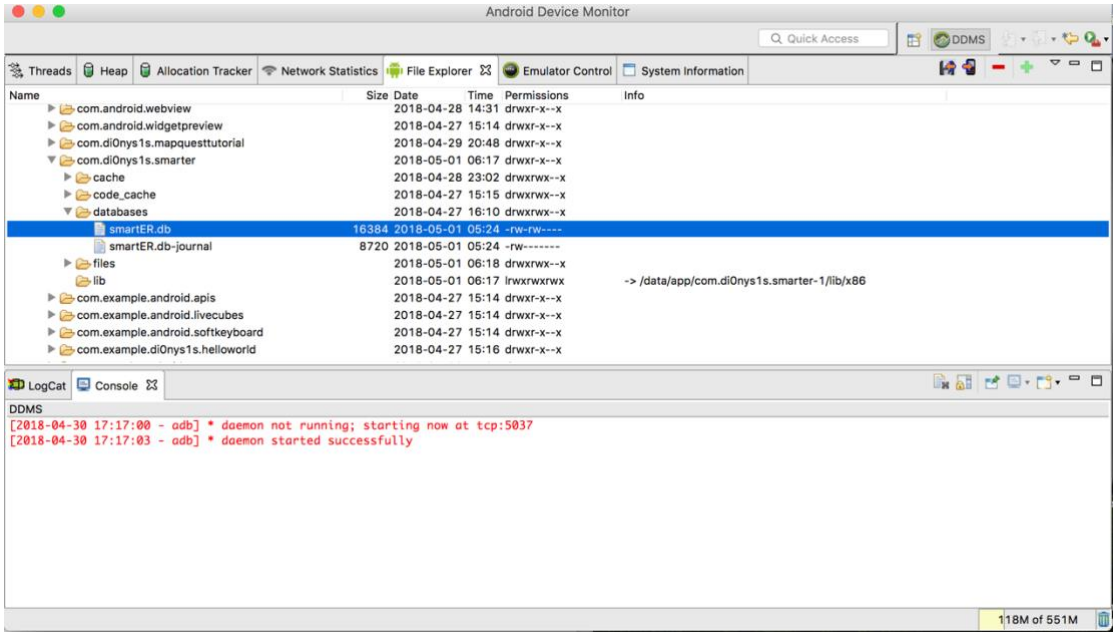
DB STRUCTURE



DB DATA

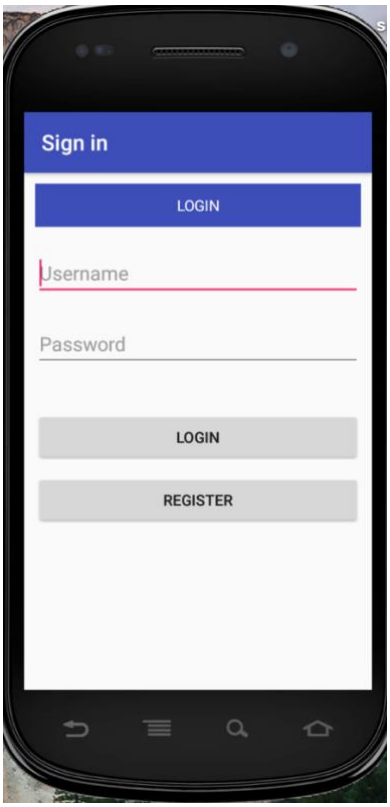


4.2 - AVD

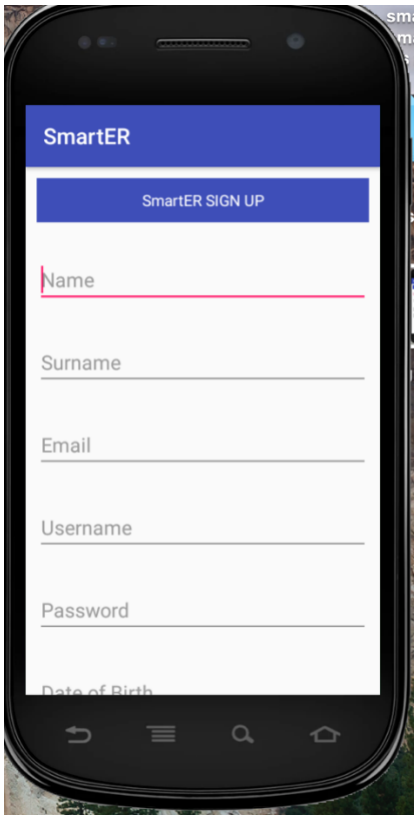

Android Device Monitor
Quick Access
DDMS
Threads
Heap
Allocation Tracker
Network Statistics
File Explorer
Emulator Control
System Information
Name Size Date Time Permissions Info
com.android.webview 2018-04-28 14:31 drwxr-x--x
com.android.widgetpreview 2018-04-27 15:14 drwxr-x--x
com.di0nys1s.mapquesttutorial 2018-04-29 20:48 drwxr-x--x
com.di0nys1s.smarter 2018-05-01 06:17 drwxr-x--x
cache 2018-04-28 23:02 drwxrwx--x
code_cache 2018-04-27 15:15 drwxrwx--x
databases 2018-04-27 16:10 drwxrwx--x
smartER.db 16384 2018-05-01 05:24 -rw-rw----
smartER.db-journal 8720 2018-05-01 05:24 -rw-------
files 2018-05-01 06:18 drwxrwx--x
lib 2018-05-01 06:17 lrwxrwxrwx -> /data/app/com.di0nys1s.smarter-1/lib/x86
com.example.android.apis 2018-04-27 15:14 drwxr-x--x
com.example.android.livecubes 2018-04-27 15:14 drwxr-x--x
com.example.android.softkeyboard 2018-04-27 15:14 drwxr-x--x
com.example.di0nys1s.helloworld 2018-04-27 15:16 drwxr-x--x
LogCat
Console
DDMS
[2018-04-30 17:17:00 - adb] * daemon not running; starting now at tcp:5037
[2018-04-30 17:17:03 - adb] * daemon started successfully
118M of 551M

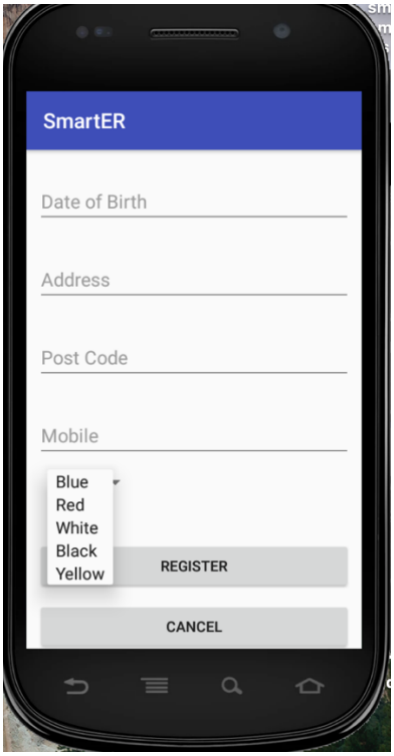
Task 5 – Application

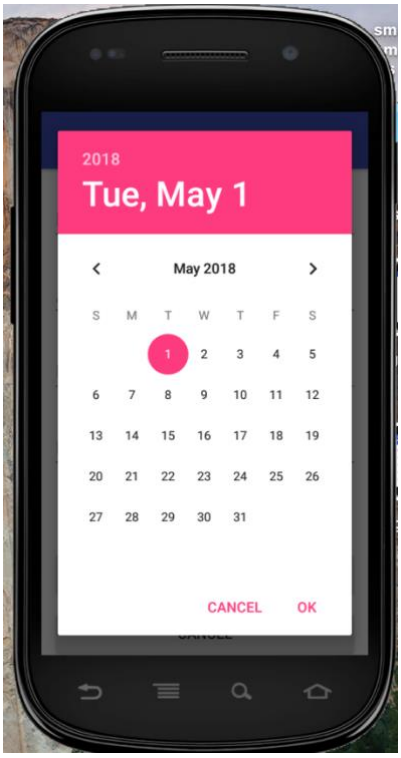
LOGIN



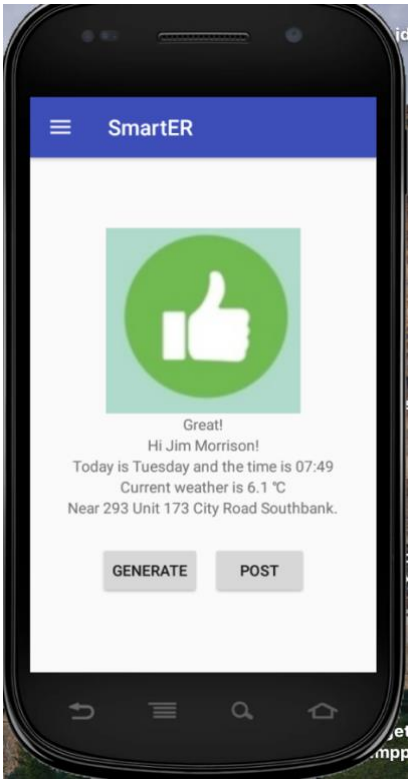
REGISTER


SmartER
SmartER SIGN UP
Name
Surname
Email
Username
Password

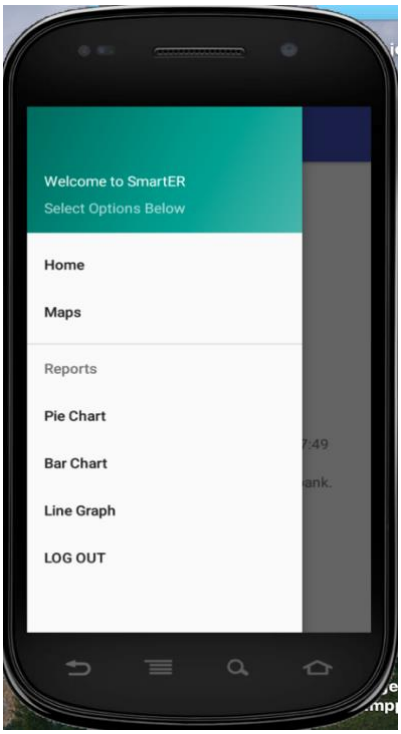

SmartER
Date of Birth
Address
Post Code
Mobile
Blue
Red
White
Black
Yellow
REGISTER
CANCEL


2018
Tue, May 1
May 2018
S M T W T F S
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
CANCEL
OK

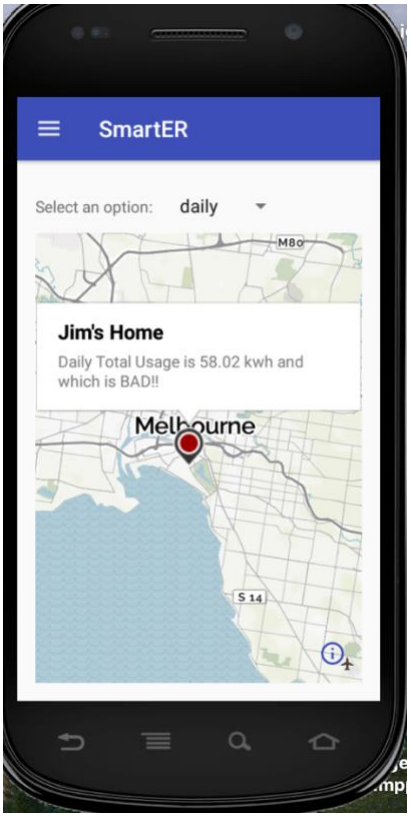
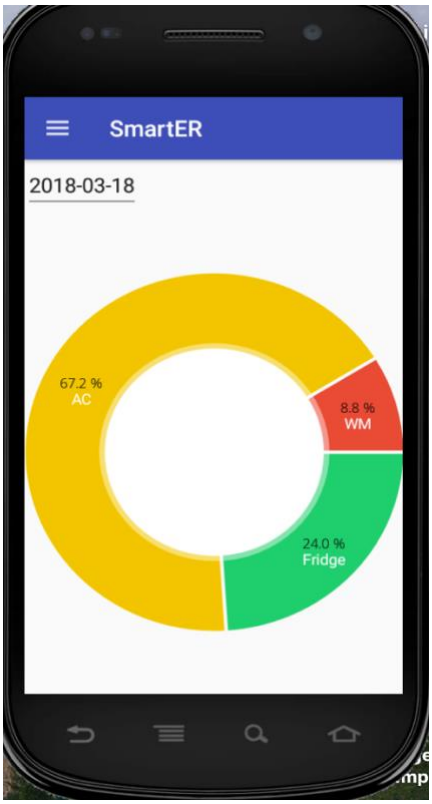
HOME


SmartER
Great!
Hi Jim Morrison!
Today is Tuesday and the time is 07:49
Current weather is 6.1 ℃
Near 293 Unit 173 City Road Southbank.
GENERATE
POST

NAVIGATION



MAP



PIE CHART