

DIABETICS SURVEY ANDROID APPLICATION

TEAM MEMBER'S

Harijeyanth k

Augustine Ajaykumar

Hariprasanth

Visvanathan

DESCRIPTION

The Diabetes Survey App is an interactive mobile application built using Android's Jetpack Compose UI toolkit. It is designed to help users provide valuable information related to their diabetes management, lifestyle, symptoms, and understanding of the condition. The app aims to make data collection simple and user-friendly while showcasing Compose's capabilities in creating dynamic UIs.

The survey covers multiple categories, including personal health information, lifestyle habits, symptom tracking, and diabetes knowledge. Users can navigate through a series of questions using an intuitive interface, with features like real-time form validation, progress tracking, and the ability to review answers before submission.

The app utilizes efficient data management, storing responses locally for privacy or syncing with a cloud service if required. Its user-centric design ensures accessibility and seamless interactions, while offering feedback or insights based on survey responses, helping users monitor their health habits and knowledge. The Diabetes Survey App is not only a demonstration of Compose's strengths but also a tool to raise awareness about diabetes and promote better health management practices.

Main Activity.java:

```
package com.app.joe.mwsleeptracker;

import android.app.ProgressDialog;
import android.content.*;
//import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.os.IBinder;
//import android.support.design.widget.Snackbar;
import android.support.v4.app.Fragment;
import android.support.v7.app.AlertDialog;
import android.util.Log;
import android.view.View;
import android.support.design.widget.NavigationView;
import android.support.v4.view.GravityCompat;
import android.support.v4.widget.DrawerLayout;
import android.support.v7.app.ActionBarDrawerToggle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
```

```
import android.view.Menu;
import android.view.MenuItem;
import android.support.v4.app.FragmentManager;
import android.support.v4.app.FragmentTransaction;
```

```
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothManager;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.CompoundButton;
import android.widget.LinearLayout;
import android.widget.Switch;
import android.widget.TextView;
//import android.widget.Toast;
//import android.widget.ToggleButton;
```

```
import com.mbientlab.metawear.AsyncOperation;
```

```
import com.mbientlab.metawear.Message;
import com.mbientlab.metawear.MetaWearBleService;
import com.mbientlab.metawear.MetaWearBoard;
import com.mbientlab.metawear.RouteManager;
import com.mbientlab.metawear.UnsupportedModuleException;
import com.mbientlab.metawear.data.CartesianFloat;
//import com.mbientlab.metawear.module.Accelerometer;
//import com.mbientlab.metawear.module.Bma255Accelerometer;
import com.mbientlab.metawear.module.Bmi160Accelerometer;

/**
 * MainActivity
 *
 * This is the activity that is started when the application is run. It creates the navigation view
 * and displays the status and info fragments.
 *
 *
 */
```



```
public class MainActivity extends AppCompatActivity implements ServiceConnection,  
NavigationView.OnNavigationItemSelectedListener {
```

```
    private MetaWearBleService.LocalBinder serviceBinder;  
    private String deviceMACAddress = "";  
    private MetaWearBoard mwBoard;  
    private ProgressDialog connectDialog;
```

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

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

```
        //Setup navigation drawer  
        DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
```

```
ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(  
this, drawer, toolbar, R.string.navigation_drawer_open, R.string.navigation_drawer_close);  
drawer.setDrawerListener(toggle);
```

```
toggle.syncState();
```

```
NavigationView navigationView = (NavigationView) findViewById(R.id.nav_view);
```

```
if (navigationView != null) {  
    navigationView.setNavigationItemSelectedListener(this);  
}
```

```
//Read the selected MW board MAC
```

```
PrefManager.Init(this);
```

```
deviceMACAddress = PrefManager.readMACAddress();
```

```
//If one has not been selected, Hide the connection switch and show that no
```

```
//device has been selected
```

```
if (deviceMACAddress == null || deviceMACAddress == ""){
```

```
Switch switchConnection = (Switch) findViewById(R.id.switchConnection);  
switchConnection.setVisibility(View.GONE);
```

```
TextView tvSelectedDevice = (TextView) findViewById(R.id.tvSelectedDevice);  
tvSelectedDevice.setText(R.string.no_device_selected);
```

```
TextView tvBoardStatus = (TextView) findViewById(R.id.tvBoardStatus);  
tvBoardStatus.setText("");
```

```
}
```

```
else{
```

```
//Otherwise, display the switch and create a listener that will detect when the  
//switch has changed states
```

```
Switch switchConnection = (Switch) findViewById(R.id.switchConnection);  
switchConnection.setVisibility(View.VISIBLE);
```

```
if (switchConnection != null) {  
    switchConnection.setOnCheckedChangeListener(new  
    CompoundButton.OnCheckedChangeListener() {
```



```
@Override
public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
    if (isChecked) {
        //If the switch is on, then connect the MW board
        connectMWBoard();

    } else {
        //If the switch is off, disconnect the MW board
        disconnectMWBoard();
    }
}
});
```

```
//Bind the MW bluetooth service and update that status fragment.
getApplicationContext().bindService(new Intent(this, MetaWearBleService.class), this,
BIND_AUTO_CREATE);
updateStatusFragment();
```

```
}
```

```
//Hide the info fragment until the MW board has been connected.  
hideInfoFragment();
```

```
//Create test entries in the database
```

```
/* DBHandler dbhandler = new DBHandler(MainActivity.this);  
dbhandler.createSampleData();  
int test = dbhandler.getRecordCount();*/  
}
```

```
private void updateStatusFragment(){  
//Calls method in status fragment to update the status  
FragmentManager fm = getSupportFragmentManager();  
MWStatusFragment fragment = (MWStatusFragment)  
fm.findFragmentById(R.id.status_fragment);  
fragment.updateStatusInfo(mwBoard, deviceMACAddress);  
}
```

```
private void updateInfoFragment(float X, float Y, float Z){  
    //Calls method in info fragment to update the display of accel info  
    FragmentManager fm = getSupportFragmentManager();  
    MWInfoFragment fragment = (MWInfoFragment)  
fm.findFragmentById(R.id.info_fragment);  
    fragment.updateDeviceInfo(X, Y, Z);  
}
```

```
@Override  
public void onBackPressed() {  
    DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
```

```
if(drawer != null){  
    if (drawer.isDrawerOpen(GravityCompat.START)) {  
        drawer.closeDrawer(GravityCompat.START);  
    } else {  
        super.onBackPressed();
```

```
}  
}  
}
```

```
@SuppressWarnings("StatementWithEmptyBody")  
@Override  
public boolean onNavigationItemSelected(MenuItem item) {  
    // Handle navigation view item clicks here.  
    Intent intent;  
  
    int id = item.getItemId();  
  
    if (id == R.id.nav_view_history) {  
        intent = new Intent(MainActivity.this, SleepLogActivity.class);  
        startActivity(intent);  
    } else if (id == R.id.nav_settings) {  
        intent = new Intent(MainActivity.this, AppSettingsActivity.class);  
        startActivity(intent);  
    } else if (id == R.id.nav_about) {
```

```
intent = new Intent(MainActivity.this, AboutActivity.class);  
startActivity(intent);  
}
```

```
DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);  
if (drawer != null) {  
    drawer.closeDrawer(GravityCompat.START);  
}
```

```
return true;  
}
```

```
@Override  
public void onDestroy() {  
    super.onDestroy();
```

```
    if (serviceBinder != null)  
        // Unbind the service when the activity is destroyed  
        getApplicationContext().unbindService(this);  
}
```



```
@Override  
public void disconnected() {  
    if (connectDialog.isShowing()) {  
        connectDialog.dismiss();  
    }  
}
```

```
hideInfoFragment();
```

```
runOnUiThread(new Runnable(){  
    @Override  
    public void run(){  
        setConnectionSwitch(false);  
    }  
});
```

```
updateStatusFragment();  
}
```

```
@Override  
public void failure(int status, Throwable error) {  
    if (connectDialog.isShowing()) {  
        connectDialog.dismiss();  
    }  
}
```

```
connectDialog.dismiss();  
}
```

```
runOnUiThread(new Runnable(){  
    @Override  
    public void run(){  
        setConnectionSwitch(false);  
    }  
});
```

```
hideInfoFragment();
```

```
updateStatusFragment();
```

```
mwBoard.connect();  
}  
});  
}
```

```
private void hideInfoFragment(){  
    //Hide the info fragment shown on this activity
```

```
FragmentManager fm = getSupportFragmentManager();  
MWInfoFragment fragment = (MWInfoFragment)  
fm.findFragmentById(R.id.info_fragment);  
FragmentTransaction ft = fm.beginTransaction();  
ft.hide(fragment);  
ft.commit();  
}
```

```
private void showInfoFragment(){  
//Show the info fragment shown on this activity  
FragmentManager fm = getSupportFragmentManager();  
MWInfoFragment fragment = (MWInfoFragment)  
fm.findFragmentById(R.id.info_fragment);  
FragmentTransaction ft = fm.beginTransaction();  
ft.show(fragment);  
ft.commit();  
}
```

```
private void setConnectionSwitch(boolean isChecked){  
    //Change the switch state based on the input parameter  
    Switch switchConnection = (Switch) findViewById(R.id.switchConnection);  
  
    if(switchConnection != null) {  
        switchConnection.setChecked(isChecked);  
    }  
}  
  
private void unsupportedModule() {  
    //Display an alert of the module is not supported by the MW board  
    AlertDialog.Builder alertDialogBuilder = new AlertDialog.Builder(this);  
  
    alertDialogBuilder.setTitle(R.string.title_error);  
    alertDialogBuilder  
        .setMessage("Unsupported Module")  
        .setCancelable(false)  
        .create()  
        .show();  
}
```

```
private void connectMWBoard(){
//Open the connection dialog
connectDialog = new ProgressDialog(MainActivity.this);
connectDialog.setTitle(getString(R.string.title_connecting));
connectDialog.setMessage(getString(R.string.message_wait));
connectDialog.setCancelable(false);
connectDialog.setCanceledOnTouchOutside(false);
connectDialog.setIndeterminate(true);
connectDialog.setButton(DialogInterface.BUTTON_NEGATIVE,
getString(R.string.label_cancel), new DialogInterface.OnClickListener() {
@Override
public void onClick(DialogInterface dialogInterface, int i) {
mwBoard.disconnect();
}
});
connectDialog.show();
```

```
//Connect to the MetaWear board
```



```
mwBoard.connect();  
}
```

```
private void disconnectMWBoard()  
mwBoard.disconnect();
```

```
hideInfoFragment();  
}
```

```
@Override  
public void onServiceDisconnected(ComponentName componentName) {  
}
```

```
public void retrieveBoard() {  
final BluetoothManager btManager=  
(BluetoothManager) getSystemService(Context.BLUETOOTH_SERVICE);  
final BluetoothDevice remoteDevice=
```

```
btManager.getAdapter().getRemoteDevice(deviceMACAddress);
```

```
// Create a MetaWear board object for the Bluetooth Device
```

```
mwBoard= serviceBinder.getMetaWearBoard(remoteDevice);
```

```
}
```

```
private void startAccelerometer() throws UnsupportedOperationException {
```

```
    Bmi160Accelerometer bmi160AccModule=
```

```
mwBoard.getModule(Bmi160Accelerometer.class);
```

```
    bmi160AccModule.setOutputDataRate(2.f);
```

```
    bmi160AccModule.setAxisSamplingRange(3.0f);
```

```
// Route data from the chip's motion detector
```

```
bmi160AccModule.routeData().fromAxes().stream("motion").commit()
```

```
.onComplete(new AsyncOperation.CompletionHandler<RouteManager>() {
```


```
    @Override
```

```
    public void success(RouteManager result) {
```

```
result.subscribe("motion", new RouteManager.MessageHandler() {  
    @Override  
    public void process(Message msg) {  
  
        updateInfoFragment(msg.getData(CartesianFloat.class).x(),  
            msg.getData(CartesianFloat.class).y(),  
            msg.getData(CartesianFloat.class).z());  
  
        //CALL TO PHYSICS ENGINE WOULD BE HERE  
  
        Log.i("MainActivity", msg.getData(CartesianFloat.class).toString());  
    }  
});  
  
bmi160AccModule.enableAxisSampling();  
  
// Switch the accelerometer to active mode  
bmi160AccModule.start();  
}
```

Output :

12:32 5G 65



Login

Username

Password

Login

[Register](#) [Forget password?](#)

☰ □ ◀

12:33 5G 65

Survey on Diabetics

Name :

Age :

Mobile Number :

Gender :

☐ Male

☐ Female

☐ Other

Diabetics :

☐ Diabetic