

DEVELOPERS DICTIONARY

APPLICATION NAME: PRIME_MRBS
DEVELOPMENT ENVIRONMENT: ANDROID STUDIO 4.1.3
GRADLE VERSION: 3.4.1
COMPILE SDK VERSION: 26
BUILD VERSION: 28.0.3
TARGET SDK VERSION: 26
MINIMUM SDK VERSION: 17

IMPLEMENTED DEPENDENCIES:

These implementations allows the use of certain features in the Android Application:

```
implementation 'com.android.support:appcompat-v7:26.+'
implementation 'com.android.support:support-v4:26.+'
implementation 'com.android.support:design:26.+'
implementation 'com.android.support:recyclerview-v7:26.+'
implementation 'com.android.support:cardview-v7:26.+'
implementation 'com.android.support.constraint:constraint-layout:1.0.2'
implementation 'com.github.simbiose:Encryption:2.0.1'
implementation files('libs/WoosimLib261.jar')
testImplementation 'junit:junit:4.12'
```

APPLICATION PERMISSIONS:

These provide permissions to use certain features in the Android Application:

```
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.INTERNET" />
```

FRAGMENTS:

Location: *App/java/com/primemrbs/*

These contains the UI layout events (Coding is in Java):

BluetoothPrintService

- Contains fixed coding used to connect to a Bluetooth device. This is a default fragment and there is no need for additional coding.

DatabaseHelper

- Contains events to read and write to the sqlite database

DataModel

- Contains events that connect sets the values in the SEARCH FORM

DeviceCheck

- Contains events that occurs during pairing to a Bluetooth device. This is a default fragment and there is no need for additional coding.

DeviceList

- Contains events to display Bluetooth devices during discovery mode.

FindingsFragment

- Contains events for the FIELD FINDINGS form.

LoginActivity

- Contains events for the LOGIN form.

MainActivity

- Contains events for the MAIN form.

NewMtrFragment

- Contains events for the ADD NEW METER form.

SearchActivity

- Contains events for the SEARCH form.

SearchAdapter

- Contains events in displaying and populating the CARD VIEW in the SEARCH form.

LAYOUTS

Location: *App/res/layout/*

These are the displayed User Interface (Coding is in XML):

activity_login.xml

- This is the LOGIN FORM UI.

activity_main.xml

- This is the MAIN FORM UI.

activity_search.xml

- This is the SEARCH FORM UI.

activity_toolbar.xml

- This displays a blank toolbar where the NAVIGATION BAR will be populated in the MAIN FORM.

app_bar_main.xml and content_main.xml

- This displays and populates the MENU ITEMS in a toolbar in the top of the MAIN FORM.

device_list.xml and device_name.xml

- This is the UI for the Bluetooth list when in Bluetooth pairing/discovery mode.

Findings_fragment.xml

- This is the FIELD FINDINGS FORM UI.

Findings_row.xml

- This is the Drop down list for the FIELD FINDINGS.

itemview.xml

- This is the UI for the CARDS displayed in the SEARCH FORM.

Nav_header_main.xml

- This is the UI for the right navigation pane.

Newmtr_fragment.xml

- This is the NEW METER FORM UI.


Spinner_layout.xml

- This is the Drop down list for the USERNAMES in the LOGIN FORM UI.

WOOSIM PRINTER JAR FILE

Location: *PRIME_MRBS/app/libs/*

This is where the WOOSIM PRINTER JAR file is located and should be located for printing to work.

Name	Date modified	Type	Size
 WoosimLib261	3/12/2020 10:34 AM	Executable Jar File	17 KB

JAR FILE must then be implemented in **build.gradle** under the **Gradle Scripts**.

FORM PROCEDURES:

The listed procedures are the widely used procedures in the MRBS. Other procedures are explained as comments in the source code itself.

```
//PROCEDURE TO JUMP TO THE LAST RECORD//
void NavLast()
//PROCEDURE TO JUMP TO THE FIRST RECORD//
void NavFirst()
//PROCEDURE TO GO TO THE NEXT RECORD//
void NavNext()
//PROCEDURE TO GO TO THE PREVIOUS RECORD//
void NavPrev()
//PROCEDURE TO GO TO THE PREVIOUS UNREAD RECORD//
void NavPrevUnread()
//PROCEDURE TO GO TO THE NEXT UNREAD RECORD//
void NavNextUnread()
//ADDITIONAL PROCEDURE USED TO REFRESH THE DATA//
public void RefreshData()
//CAMERA LOADING PROCEDURE//
public void CameraFunc()
//ENTER PRESENT READING PROCEDURE//
public void EnterRdg()
//INDICATION FOR SENIOR CITIZEN CHARGE VALUE FOR SENIOR CITIZEN CONSUMERS WITH
LESS THAN 30 CUM//
public void ComputeSCdisc()
//INDICATION FOR SENIOR CITIZEN CHARGE VALUE FOR SENIOR CITIZEN CONSUMERS WITH
MORE THAN 30 CUM//
public void ComputeSCdiscAfterMax()
//PROCEDURE FOR CHARGES COMPUTATION//
public void ComputeBill()
//PROCEDURE FOR LOADING THE FIELD FINDINGS FORM//
public void OpenRemarks()
//PROCEDURE TO SAVE READINGS IN THE DATABASE//
public void SaveReadings()
//PROCEDURE TO RELOAD READ AND UNREAD STATUS FOR STATISTICS//
public void CheckStatus()
//PROCEDURE TO LOAD ALL CONSUMER DATA FROM THE DATABASE TO THE VARIABLES OF THIS
APPLICATION//
public void GetSQLiteDatabaseRecords()
//GPS CONDITIONS//
private void buildAlertMessageNoGps()
private void buildAlertMessageHaveGps()
void getLocation()
//PROCEDURE TO LOAD OTHERS DATA FROM THE DATABASE TO THE VARIABLES OF THIS
APPLICATION//
public void getOthers()
//PROCEDURE TO LOAD WATER RATES DATA FROM THE DATABASE TO THE VARIABLES OF THIS
APPLICATION//
public void GetSQLiteDatabaseWRates()
//PROCEDURE FOR PRINTING, WITH MESSAGEBOX FOR CONFIRMATION//
public void PrintAsk()
//PROCEDURE FOR SETTING UP BARCODE DATA//
public void SetupBarcodes()
//PROCEDURE FOR SOA PRINTING//
public void printSOA()
//PROCEDURE FOR PRINTING READ RECORDS//
public void printRead()
```

```
//PROCEDURE FOR PRINTING RECORDS 1 TO 100//  
public void print1to100() throws IOException  
//PROCEDURE FOR PRINTING RECORDS 101 TO 200//  
public void print101to200() throws IOException  
//PROCEDURE FOR PRINTING RECORDS 200 AND UP//  
public void print201up() throws IOException  
//PROCEDURE FOR LOADING IMAGE FOR SOA PRINTING//  
private void sendImg(int x, int y, int id)  
//PROCEDURE FOR STORAGE PERMISSIONS VERIFICATION//  
public static void verifyStoragePermissions(Activity activity)
```

IMPORTANT NOTES:

- Always make sure that DEVELOPER OPTIONS is enabled during debugging. If DEVELOPER OPTIONS is not activated, you will not be able to debug in a connected physical device.
- Under DEVELOPER OPTIONS, USB DEBUGGING should also be ENABLED.
- The SQLITE database location should be in a folder named “databases” in the ROOT FOLDER of the device.