**Data Storage**

Description:

Android allows developer to store local data in the phone using different format (files, Shared references, SQLite). This data will store in same path that application will store in the phone. it is:

**“data/data/package\_name”**

if we explore this path we will see three folders that have all local data storage

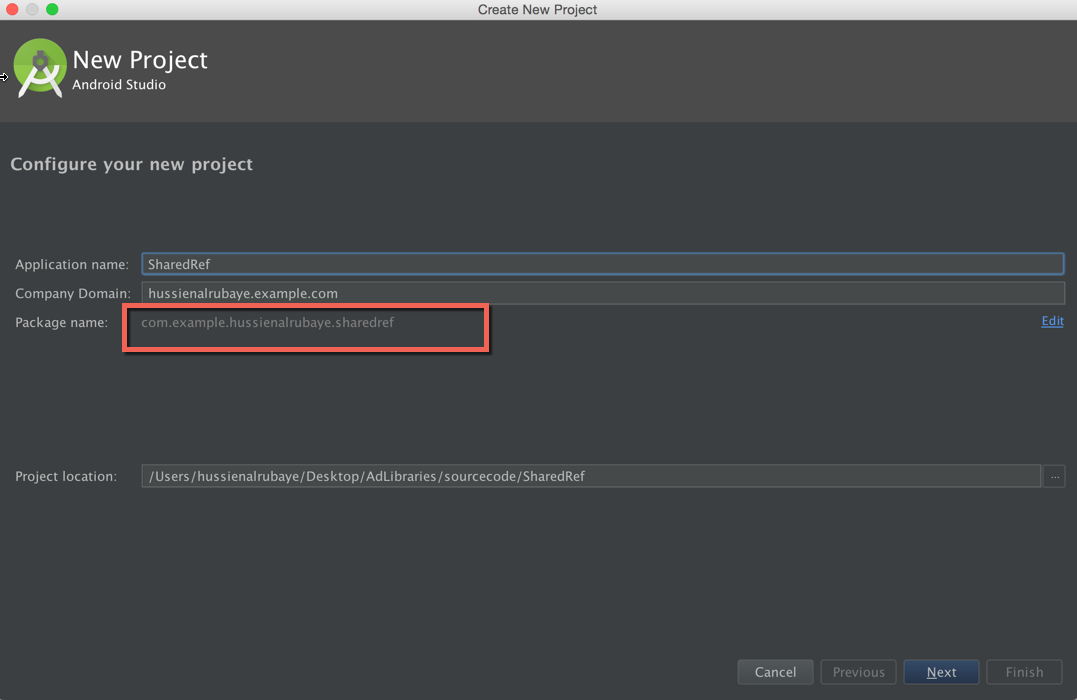
* Files
* Databases
* Shared Ref

The important thing to know that Android did not encrypt this data so anyone or any app could go to that path and read data stored in your (files, Shared references, SQLite.

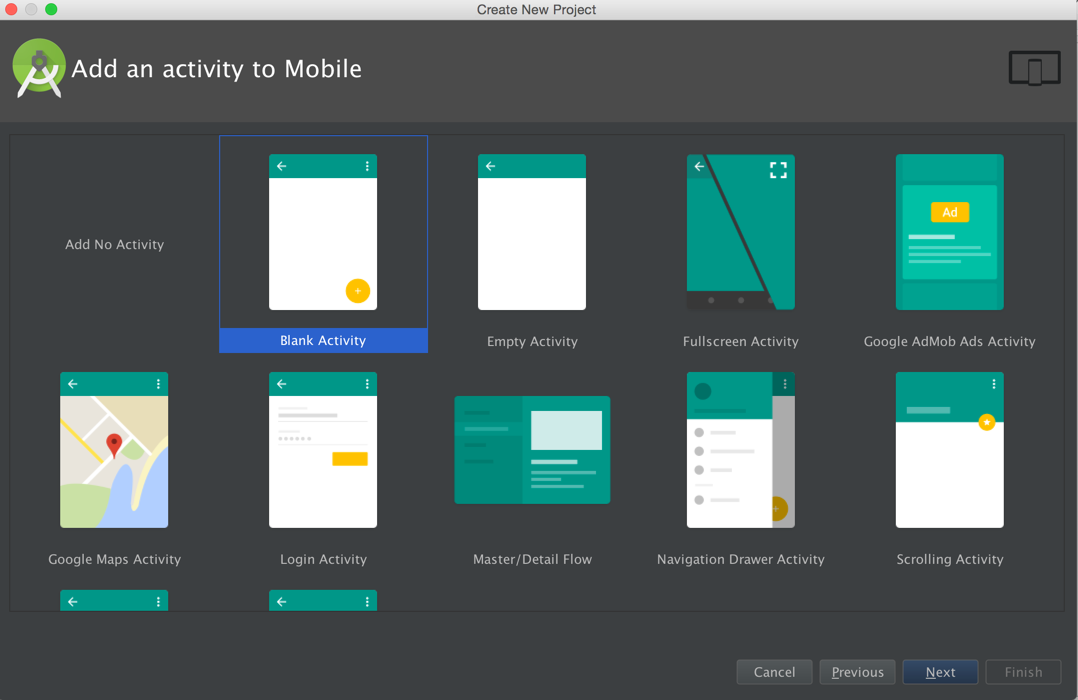
So we advise that don’t use these files to store secure information like (user name and password). We will build simple app that store user name and password in Shared references then we will show how this data could be access and read

**Steps to build the app**

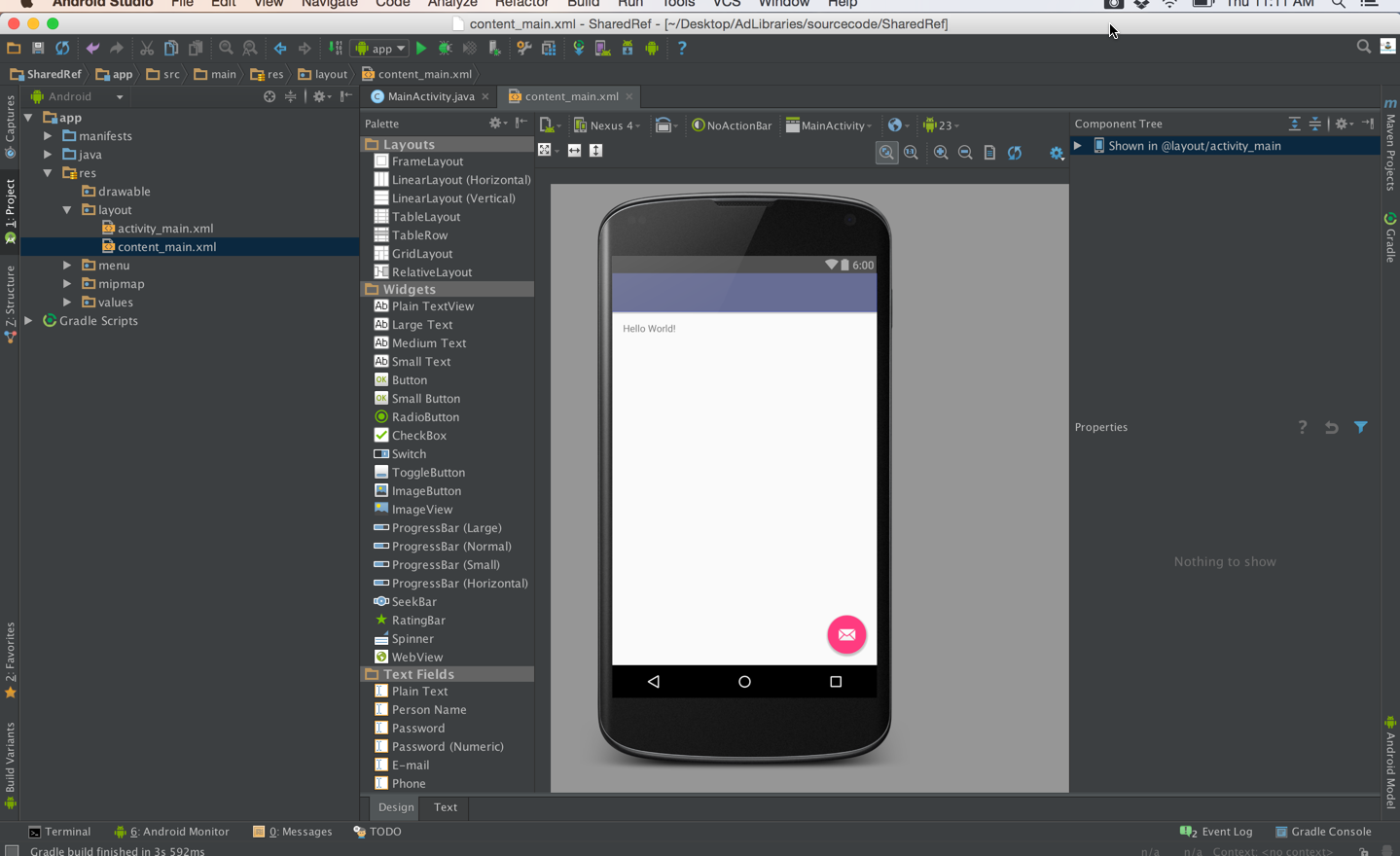
1. Open new project with name “shared Ref”, save the package name will will need next



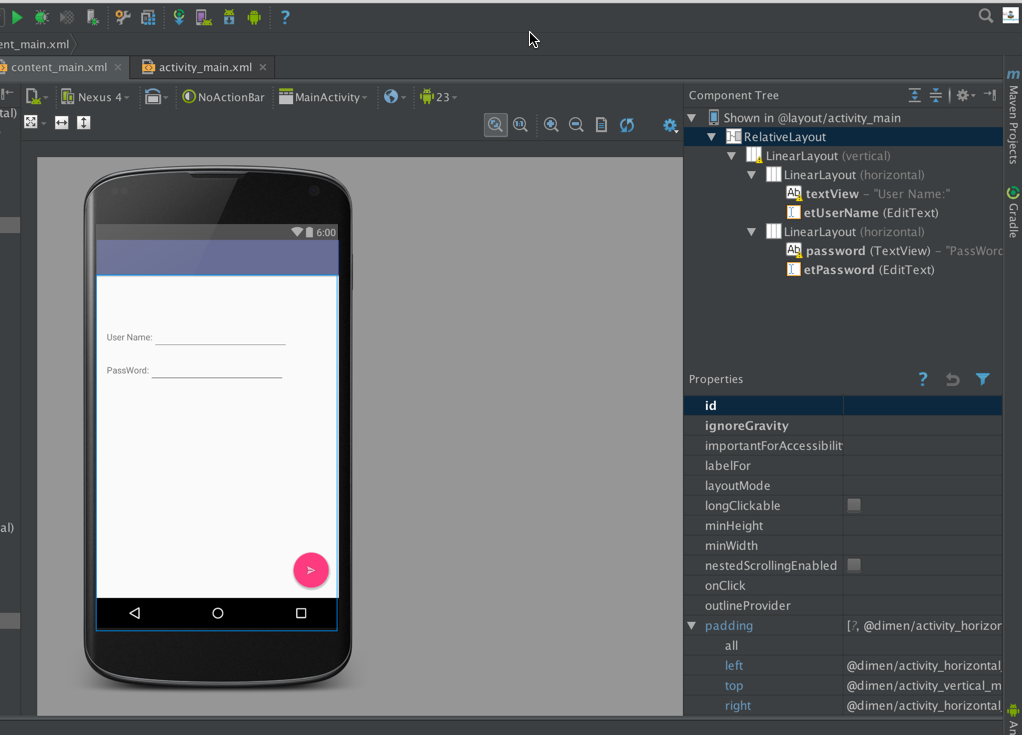
1. Select blank activity



We will see the app like this



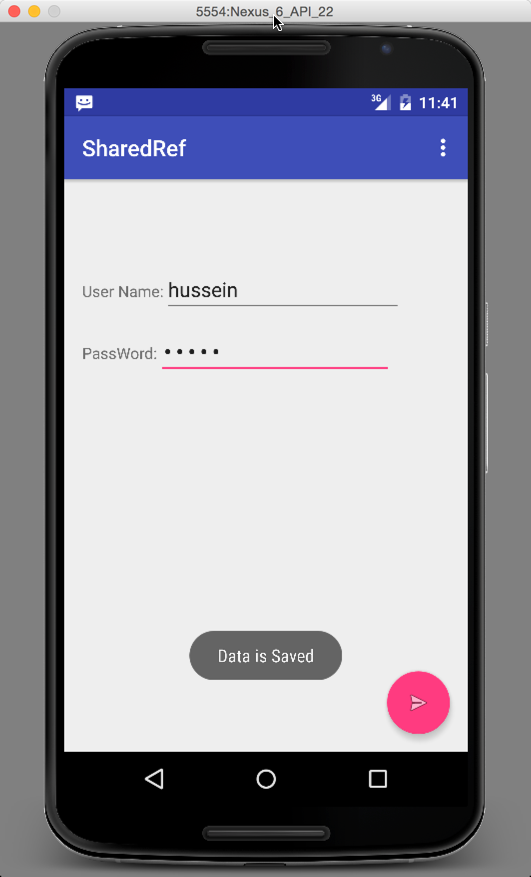
1. We will add some objects ( TextView, EditText) and make the app like this, see the name of every tool in the right.



1. The code will be like this code

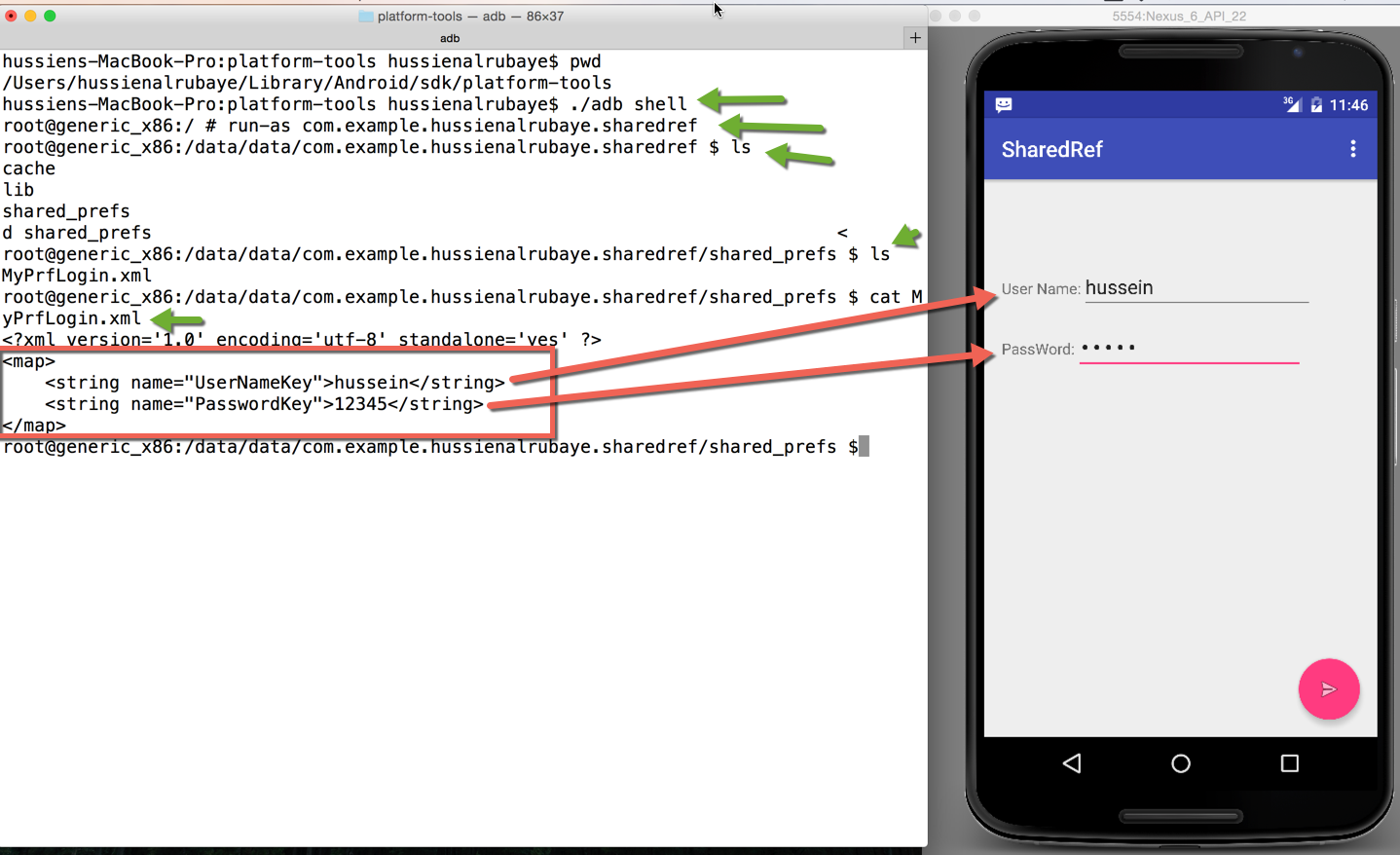
|  |
| --- |
| Java |
| // shared references files name public static final String *MyPREFERENCES* = "MyPrfLogin" ; // key for user name public static final String *UserName* = "UserNameKey"; // key for password public static final String *Password* = "PasswordKey"; // shared references instance to access to virtual file SharedPreferences sharedpreferences; // input text name EditText etUserName; // input text password EditText etPassword; @Override protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);  Toolbar toolbar = (Toolbar) findViewById(R.id.*toolbar*);  setSupportActionBar(toolbar);  // initialize user name instance with the real input in xml  etUserName=(EditText)findViewById(R.id.*etUserName*);  // initialize password instance with the real input in xml  etPassword=(EditText)findViewById(R.id.*etPassword*);  // // initialize shared references  sharedpreferences = getSharedPreferences(*MyPREFERENCES*, Context.*MODE\_PRIVATE*);  // access to the floating button  FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.*fab*);  // listen to floating button when click  fab.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View view) {  // save data  // enable start editing file  SharedPreferences.Editor editor = sharedpreferences.edit();  // add user name  editor.putString(*UserName*, etUserName.getText().toString());  // add password  editor.putString(*Password*, etPassword.getText().toString());  // save the update data  editor.commit();  //display message saved  Toast.*makeText*(MainActivity.this, "Data is Saved", Toast.*LENGTH\_LONG*).show();   }}); } |

1. Run the app and add user name and password



**Steps to read the app data**

To Read the app Data, Run Terminal and execute the command as it shows below



Conclusion:

As you see we could read the same data that stored in the app, if it be sensitive information this will be very bad.

**Fix The Problem:**

To fix the problem you have have two solution

1. Encrypt sensitive data in the phone using encryption and decryption
2. save sensitive data in the server and retrieved when wee need it.

**What we will do:**

We demonstrated small app that encrypt data before save into the share references, and when we want use it we will decrypt it.

**Steps:**

We will take same app that we build before and only encrypt the data before saving it into the shared file using basic cipher

|  |
| --- |
| Java |
| // cipher encryption add shift for key String cipher(String msg, int shift){  String s = "";  int len = msg.length(); // get string length  for(int x = 0; x < len; x++){  char c = (char)(msg.charAt(x) + shift); // shift every character  s += c; // append the characters  }  return s; } |

This encryption is very basic just adding shift for every character when we want to encrypt data, and when we want decryption we add add minus that shift to origin character

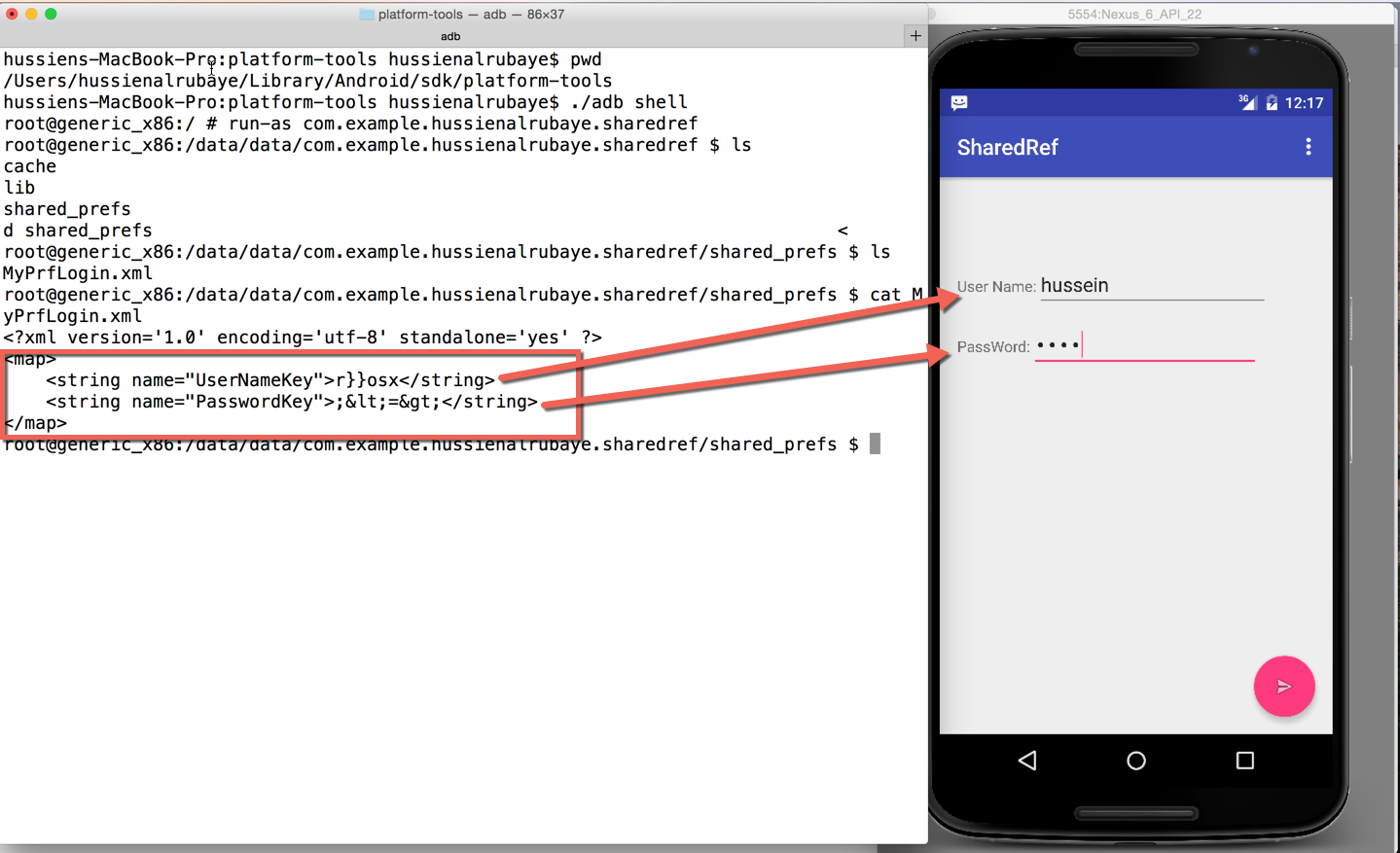
|  |
| --- |
| Java |
| // cipher encryption add shift for key  cipher(“hussein”, 10) // result “r }}osx”  // cipher Decryptions  cipher(“r }}osx”, -10) / result “Hussein” |

The code will update to

|  |
| --- |
| Java |
| // shared references files name public static final String *MyPREFERENCES* = "MyPrfLogin" ; // key for user name public static final String *UserName* = "UserNameKey"; // key for password public static final String *Password* = "PasswordKey"; // shared references instance to access to virtual file SharedPreferences sharedpreferences; // input text name EditText etUserName; // input text password EditText etPassword; @Override protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);  Toolbar toolbar = (Toolbar) findViewById(R.id.*toolbar*);  setSupportActionBar(toolbar);  // initialize user name instance with the real input in xml  etUserName=(EditText)findViewById(R.id.*etUserName*);  // initialize password instance with the real input in xml  etPassword=(EditText)findViewById(R.id.*etPassword*);  // // initialize shared references  sharedpreferences = getSharedPreferences(*MyPREFERENCES*, Context.*MODE\_PRIVATE*);  // access to the floating button  FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.*fab*);  // listen to floating button when click  fab.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View view) {  // save data  // enable start editing file  SharedPreferences.Editor editor = sharedpreferences.edit();  // add user name  editor.putString(*UserName*,cipher(etUserName.getText().toString(),10));  // add password  editor.putString(*Password*,cipher( etPassword.getText().toString(),10));  // save the update data  editor.commit();  //display message saved  Toast.*makeText*(MainActivity.this, "Data is Saved", Toast.*LENGTH\_LONG*).show();   }  }); }  // cipher encryption add shift for key String cipher(String msg, int shift){  String s = "";  int len = msg.length(); // get string length  for(int x = 0; x < len; x++){  char c = (char)(msg.charAt(x) + shift); // shift every character  s += c; // append the characters  }  return s; } |

**Steps to read the app data**

To Read the app Data, Run Terminal and execute the command as it shows bellow



See you cannot understand the data because it is encrypted.