```
class LoginActivity : Activity() {
    lateinit var input : EditText
    lateinit  var  password : EditText
    lateinit var dbHelper : DBHelper
    var passwordEncrypter = PasswordEncryptor()

    companion object {
        var sharedPreferenceFile : String = "ShareWithPreference"
    }

    override fun onCreate(savedInstanceState: Bundle?) {

        super.onCreate(savedInstanceState)
       setContentView(R.layout.activity_login)

        dbHelper = DBHelper(this,null)
        val signupLink = findViewById<TextView>(R.id.signup_link) as TextView
        val loginButton = findViewById<Button>(R.id.login_button) as Button

        signup_link?.setOnClickListener{
            val intent = Intent(this, SignupActivity::class.java)
            startActivity(intent)

        }
        login_button?.setOnClickListener{
            processLoginDetails()
        }
    }


fun processLoginDetails()
    {
        input = findViewById<EditText>(R.id.input_user) as EditText
        password = findViewById<EditText>(R.id.input_password) as EditText

        if(input.text.toString()!= "" && password.text.toString() != "")
        {
         if(isEmailValid(input.text.toString()))
         {
            val cursor : Cursor =
dbHelper.getUserByEmail(input.text.toString(),pswdEnc.encodeData(password.text.toSt
ring()))
            try {
               if (cursor!!.moveToFirst()) {

                  if (cursor == null || cursor.count == -1) {
                     Log.e("Bye", "User not found by email")
                     Toast.makeText(this,"Check Username and
password",Toast.LENGTH_SHORT).show()
                     input.text.clear()
                     password.text.clear()
                  } else {
                     val sharedPreference: SharedPreferences =
this.getSharedPreferences(sharedPreferanceFile, Context.MODE_PRIVATE)
                     val currentUserEmailId =
cursor.getString(cursor.getColumnIndex(DBHelper.COLUMN_EMAILID))
                     val currentUserMobileNumber =
cursor.getString(cursor.getColumnIndex(DBHelper.COLUMN_MOBILE))
                     val currentUserName =
cursor.getString(cursor.getColumnIndex(DBHelper.COLUMN_NAME))
```
/*NOTE :
For variables,

***/**
```
                     val editor: SharedPreferences.Editor = sharedPref.edit()
                     editor.putString("EMAIL_KEY", sharedEmail)
                     val editor:SharedPreferences.Editor =  shsaredPreference.edit()
                     editor.putString("EMAIL_KEY",currentUserEmailId)
                     editor.putString("MOBILE_KEY",currentUserMobileNumber)
                     editor.putString("NAME_KEY",currentUserName)

                     editor.apply()
                     editor.commit()
                     val intent = Intent(this, TabbedMainActivity::class.java)
```

```
                    startActivity(intent)
                    finish()

                  }
                }
                }
                catch(e : Exception) {
                  Log.e("exception in login  ", "  in catch $e")
                }
            }

            else if(isMobileNumber(input.text.toString()))
            {
              val cursor : Cursor =
dbHelper.getUserByMobile(input.text.toString(),pswdEnc.encodeData(password.text.toS
tring()))
              try{
                if(cursor!!.moveToFirst())
                {
                  if(cursor == null || cursor.count == -1)
                  {
                    Log.e("Bye","User not found by mobile")
                    //Toast.makeText(this,"")
                  }
                  else{
                    val sharedPreference : SharedPreferences =
this.getSharedPreferences(sharedPreFile, Context.MODE_PRIVATE)
                    val currentUserEmailId =
cursor.getString(cursor.getColumnIndex(DBHelper.COLUMN_EMAILID))
                    val currentUserMobileNumber =
cursor.getString(cursor.getColumnIndex(DBHelper.COLUMN_MOBILE))
                    val currentUserName =
cursor.getString(cursor.getColumnIndex(DBHelper.COLUMN_NAME))

                    val editor:SharedPreferences.Editor =  shsaredPreference.edit()
                    editor.putString("EMAIL_KEY",currentUserEmailId)
                    editor.putString("MOBILE_KEY",currentUserMobileNumber)
```

```
                        editor.putString("NAME_KEY",currentUserName)
                        editor.apply()
                        editor.commit()
                        val intent = Intent(this, TabbedMainActivity::class.java)
                        startActivity(intent)
                        finish()
                    }
                }
            }
            catch(e : Exception){
                Log.e("exception ", " $e")
            }
        }
    }
}


fun isValidEmailId(email: String): Boolean {
    return android.util.Patterns.EMAIL_ADDRESS.matcher(email).matches()
}

fun isMobileNumber(mobile : String) : Boolean{
    return android.util.Patterns.PHONE.matcher(mobile).matches()
}

}
```

The functions processLoginDetails() has 50+ Lines of Code, so we can split up the code into smaller groups.
This function process login logic based on whether the user login using mobile number or emailId, so we can split up the functionality as two separate functions. Also the current user storing process in shared preference is repeated in both of the conditions, so we also add another function as #storeCurrentUserDetails().

After Modification:

```
fun processLoginDetails() {
      input = findViewById<EditText>(R.id.input_user) as EditText
      password = findViewById<EditText>(R.id.input_password) as EditText

      if(input.text.toString()!= "" && password.text.toString() != "")
      {
         val userId = input.text.toString()
         val password = password.text.toString()
         if(isEmailValid(userId)){
                  processLoginDetailsByEmailId(userId, password)
         }else if(isMobileNumber(userId)){
                  processLoginDetailsByMobileNumber(userId, password)
         }
         }
 }

fun processLoginDetailsByEmailId(userId: String, password : String){
         val cursor : Cursor =
dbHelper.getUserByEmail(userId,passwordEncryptor.encodeData(password))
            try {
               if (cursor!!.moveToFirst()) {

                  if (cursor == null || cursor.count == -1) {
                     Log.e("Bye", "User not found by email")
                     Toast.makeText(this,"Check Username and
password",Toast.LENGTH_SHORT).show()

                  } else {
                           val currentUserEmailId =
cursor.getString(cursor.getColumnIndex(DBHelper.COLUMN_EMAILID))
                     val currentUserMobileNumber =
cursor.getString(cursor.getColumnIndex(DBHelper.COLUMN_MOBILE))
                     val currentUserName =
cursor.getString(cursor.getColumnIndex(DBHelper.COLUMN_NAME))
                  storeCurrentUserDetails(currentUserEmailId,
currentUserMobileNumber,currentUserName)
                  val intent = Intent(this, TabbedMainActivity::class.java)
```

```
                     startActivity(intent)
                     finish()
                    }
          }
          }
}

fun processLoginDetailsByMobileNumber(userId: String, password : String){
          val cursor : Cursor =
dbHelper.getUserByMobile(userId,passwordEncryptor.encodeData(password))
             try {
                if (cursor!!.moveToFirst()) {

                    if (cursor == null || cursor.count == -1) {
                       Log.e("Bye", "User not found by email")
                       Toast.makeText(this,"Check Username and
password",Toast.LENGTH_SHORT).show()

                    } else {
                              val currentUserEmailId =
cursor.getString(cursor.getColumnIndex(DBHelper.COLUMN_EMAILID))
                       val currentUserMobileNumber =
cursor.getString(cursor.getColumnIndex(DBHelper.COLUMN_MOBILE))
                       val currentUserName =
cursor.getString(cursor.getColumnIndex(DBHelper.COLUMN_NAME))
                    storeCurrentUserDetails(currentUserEmailId,
currentUserMobileNumber,currentUserName)
                    val intent = Intent(this, TabbedMainActivity::class.java)
                     startActivity(intent)
                     finish()
                    }
          }
          }
}
```

```
fun storeCurrentUserDetails(emailId : String, mobileNumber: String, name : String){
        val editor:SharedPreferences.Editor =  sharedPreference.edit()
                editor.putString("EMAIL_KEY",emailId)
                editor.putString("MOBILE_KEY",mobileNumber)
                editor.putString("NAME_KEY",name)
                editor.apply()
                editor.commit()

}
```

References :
https://www.freecodecamp.org/news/clean-coding-for-beginners/

https://medium.com/swlh/clean-code-writing-functions-or-methods-4e6e53ff4ac2

https://www.fabrizioduroni.it/2019/07/28/clean-code-functions/

https://workat.tech/machine-coding/tutorial/design-good-functions-classes-clean-code-86h68awn9c7q

https://www.todaysoftmag.com/article/1071/clean-code-functions

https://dipta007.com/clean-code-functions/