Assignment 1

Deadline: Day 04/03/2018 @ 23:59

**[Total Mark for this Assignment is 20/5=4]**

***Mobile Application Development***

***IT448***

**Instructions:**

* This Assignment must be submitted on Blackboard (**WORD format only**) via the allocated folder.
* Email submission will not be accepted.
* You are advised to make your work clear and well-presented, marks may be reduced for poor presentation. This includes filling your information on the cover page.
* You MUST show all your work, and text must not be converted into an image, unless specified otherwise by the question.
* Late submission will result in ZERO marks being awarded.
* The work should be your own, copying from students or other resources will result in ZERO marks.
* Use **Times New Roman** font for all your answers.

Student Details:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Name:** ###**CRN:** ### |  | **ID:** ### |
|  |  |  |

# Question One

***1 Marks***

*Learning Outcome(s):*

*Understand the impacts of mobile technology.*

1. Discuss how mobile applications are changing business processes?

Processes are designed within the parameters of the available technology. When technology drastically changes, new forms are enabled. When that technology is cheap, change is enabled in areas that may have previously seen limited impact of the technology.

Businesses are paying significant attention to mobile because these qualities suggest that the technology may have implications for strategic and tactical advantage. This is very beneficial especially in smaller businesses that may have found that the cost, complexity, and non-mobile nature of traditional computing platforms made technological solutions to their business process infeasible.

Large organizations can also benefit from process redesign based on mobile technology, and many are creating mobile development teams to explore, design, and implement process solutions.

The focus for these companies is on internal processes, and they are large enough to absorb the cost of creating apps to support their processes. These apps are generally not available to other companies via an app market, although some apps available to consumers hint at the internal process changes. One such example is the insurance company apps that allow customers to provide insurance claims.

# Question Two

***2 Marks***

*Learning Outcome(s):*

Identify and compare technologies that enable the development of applications for mobile devices.

* 1. What is the official IDE for Android app development and list some important features of that IDE that enhance your productivity when building Android apps?
	2. Describe the package explorer in Android app development IDE.

Answer:

**Note: student list may vary for the IDE features list. They can use the same features mentioned in the slides or something like the below answer taken from online sources.**

1. Android Studio is the official IDE for android app development. The important features of android studio IDE are listed below
* A unified environment where you can develop for all Android devices
* Code templates and GitHub integration to help you build common app features and import sample code
* Extensive testing tools and frameworks
* Lint tools to catch performance, usability, version compatibility, and other problems
* C++ and NDK support
* A flexible Gradle-based build system
* A fast and feature-rich emulator
* Instant Run to push changes to your running app without building a new APK
* Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine

# Question Three

***7 Marks***

*Learning Outcome(s):*

Design application interfaces for mobile devices using appropriate software and/or programming.

1. Create a new Android project called Assignment1.
	1. Set the application name to **YourName**. Create the first Activity and name it as **YourName\_LoginActivity**.
	2. In the LoginActivity, design a login form to input user name and password as well as a LOGIN button as displayed in image below. The user can input his/her name and password. ***Suppose the required username =”sa” and required password= “sa”***

Wrong username or password



* 1. Create a second Activity called **YourName\_SecondActivity**. When the user clicks on LOGIN button, compare the entered username and password with **“sa”**. If right, the SecondActivity should be started through an intent. The intent should carry first activity name **or** username and display it in a text view as shown in figure above. If password or username is wrong, a red label appears on the first activity stating “wrong username or password” (screenshot 3)

 **Requirement:**

**Attach both xml and java code for your activities.**

**Answer:**

package com.seu.ABC;

import android.content.Intent;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

public class Login extends AppCompatActivity

{

private static EditText username;

private static EditText password;

private static Button login\_button;

private static TextView txtMessage;

@Override

protected void onCreate(Bundle savedInstanceState)

{

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_login);

LoginButton();

}

public void LoginButton()

{

username = (EditText)findViewById(R.id.editText\_user);

password = (EditText)findViewById(R.id.editText\_password);

login\_button = (Button)findViewById(R.id.button\_login);

login\_button.setOnClickListener(

new View.OnClickListener()

{

@Override

public void onClick(View v) {

if((username.equals(“sa”) && (password.equals(“sa”)){

txtMessage.setVisibility(View.***INVISIBLE***);

Intent intent = new Intent(LoginActivity.this, SecondActivity.class);

intent.putExtra(“username”, username)

startActivity(intent);

}

else{

txtMessage= (TextView) findViewById(R.id.txtMessage);

txtMessage.setVisibility(View.***VISIBLE***);

}

}

}

);

}

}

**public class Login extends AppCompatActivity**

{

 @Override

protected void onCreate(Bundle savedInstanceState)

{

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_second);

 Bundle extras= getIntent().getExtras();

 String username=extras.getString(**"**username**"**);

TextView txtDisplay= (TextView)findViewById(R.id.txtDisplay);

 txtDisplay.setText(**"**Welcome **"** + username);

}

}

# Question Four

***3 Marks***

*Learning Outcome(s):*

Design application interfaces for mobile devices using appropriate software and/or programming.

1. Creating a functional custom complex list in Android includes accomplishing three steps: create data source methods, create custom list item layout, and create custom Adapter.
	1. Explain why creating the custom Adapter is required instead of using the default ArrayAdapter.
	2. State the required changes needed to test the custom adapter that holds “Student” objects.

PS: suppose the following:

* activity name is “StudentsListActivity”
* custom adapter name is “StudentAdapter”
* The data source method used to return the Student objects is called: ds.getStudents();

Answer:

1. Creating a custom complex list without creating a custom adapter is almost impractical. Custom adapters are always created as subclasses of another type of adapter (ArrayAdapter) that would provide the desired behavior and that has been parameterized to hold only Item objects.
2. The following changes are required to test the customer adapter:
* change the StudentsListActivity code to retrieve Item objects rather than Item names,
	+ *“final ArrayList<Student> students = ds.getStudents();”*
* set the ListView to use the custom adapter
	+ **“***setListAdapter(* ***new*** *StudentAdapter(* ***this*** *, Students));*”

# Question Five

***7 Marks***

*Learning Outcome(s):*

Design application interfaces for mobile devices using appropriate software and/or programming.

Consider that you are a developing an app for your university that stores the contacts details for all faculty members.

1. Complete the DatabaseHandler class
	* Complete the missing code (1) & (2)
	* Create the **Contacts table** (3). The static variables in the code represent the column names for the table to store the information.
2. Complete the DataSource class
	* Complete the missing code (4) & (5)
	* Insert a record into the **Contacts** table. Use any values for each table column (6)
	* Retrieve the contact last\_name of a specific id (passed as parameter) (7)
	* Update the contact mobile phone of a specific user\_id (8)

**Marking Criteria**:

* (1) 0.5 marks
* (2) 0.5 marks
* (3) 2 marks
* (4) 0.5 marks
* (5) 0.5 marks
* (6) 1 mark
* (7) 1 mark
* (8) 1 mark

**public class** DatabaseHandler **extends** \_\_SQLiteOpenHelper\_\_\_\_\_\_\_\_ //(1)

{
  *// Database Version* **private static final int *DATABASE\_VERSION*** = 1 ;
 *// Database Name* **private static final** String ***DATABASE\_NAME*** = **"seu"**;

 *// Contacts table name* **private static final** String ***TABLE\_CONTACTS*** = **"Contacts"**;

 *// Contacts Table Columns names* **private static final** String ***KEY\_ID*** = **"id"**;
 **private static final** String ***KEY\_FIRST\_NAME*** = **"first\_name"**;
 **private static final** String ***KEY\_LAST\_NAME*** = **"last\_name"**;
 **private static final** String ***KEY\_MOBILE\_NUMBER*** = **"mobile\_number"**;
 **private static final** String ***KEY\_EMAIL\_ID*** = **"email\_id"**;
 **private static final** String ***KEY\_USER\_ID*** = **"user\_id"**;
 **private static final** String ***KEY\_PASSWORD*** = **"password"**;
 **private static final** String ***KEY\_ROLE*** = **"user\_role"**;

 **public** DatabaseHandler(Context context) {

 **\_super** (context, *DATABASE\_NAME* , **null** , *DATABASE\_VERSION* );**\_\_\_** //(2)
 }

 *// Creating Table Contacts* @Override

 **public void** onCreate(SQLiteDatabase db) { //(3)

String CREATE\_CONTACTS\_TABLE = **"CREATE TABLE "** + ***TABLE\_CONTACTS*** + **"("** + ***KEY\_ID*** + **" INTEGER PRIMARY KEY,"** +
 ***KEY\_FIRST\_NAME*** + **" TEXT,"** +
 ***KEY\_LAST\_NAME*** + **" TEXT,"** +
 ***KEY\_MOBILE\_NUMBER*** + **" TEXT,"** +
 ***KEY\_EMAIL\_ID*** + **" TEXT,"** +
 ***KEY\_USER\_ID*** + **" TEXT,"** +
 ***KEY\_PASSWORD*** + **" TEXT,"** +
 ***KEY\_ROLE*** + **" TEXT"** +
 **")"**;
 db.execSQL(CREATE\_CONTACTS\_TABLE);
 }

}

**public class** DataSource {

 **private** SQLiteDatabase database ;

 **private** DatabaseHandler dbHelper ;

 **public** DataSource(Context context) {

 dbHelper = **new** ContactDBHelper(context); //(4)

 }

**public void** open() **throws** SQLException {

 database = dbHelper .getWritableDatabase();//(5)

}

*// Insert information into Contacts Table. Put any value for each column*
 **public boolean** addContact( ) { //(6)

 **boolean** didSucceed = **false** ;

**try** {

ContentValues values = **new** ContentValues();
 values.put(***KEY\_FIRST\_NAME***, “any value” );
 values.put(***KEY\_LAST\_NAME***, “ any value” );
 values.put(***KEY\_MOBILE\_NUMBER***, “any mob nb”);
 values.put(***KEY\_EMAIL\_ID***, “any email” );
 values.put(***KEY\_USER\_ID***, any id number);
 values.put(***KEY\_PASSWORD***, “any password”);
 values.put(***KEY\_ROLE***, “any role”);

didSucceed =db.insert(***TABLE\_CONTACTS***, **null**, values) > 0;

 }

**catch** (Exception e) {}

 **return** didSucceed;

 }

 *// Update the contact mobile number of the passed contact\_id parameter. Put any value for the updated contact mobile number*

**public boolean updateContact(int contact\_id)** // (7)

 **boolean** didSucceed = **false** ;

 **try** {

ContentValues values = **new** ContentValues();
 values.put(***KEY\_MOBILE\_NUMBER***, “any mob nb”);

 didSucceed = db .update( "contact" , values, "\_id=" + id, **null** ) > 0;

 }

**catch** (Exception e) {}

 **return** didSucceed;

}

*// Retrieve the contact last name of the passed id parameter*

**public String getContactName(int contact\_id)** // (8)**{**

String name=null;

**try** {

String query = "Select last\_name from table\_contacts where id= "+ contact\_id ;

Cursor cursor = db .rawQuery(query, **null** );

cursor.moveToFirst();

name = cursor.getString(0);

cursor.close();

}

**catch** (Exception e) { }

**return** name;

}