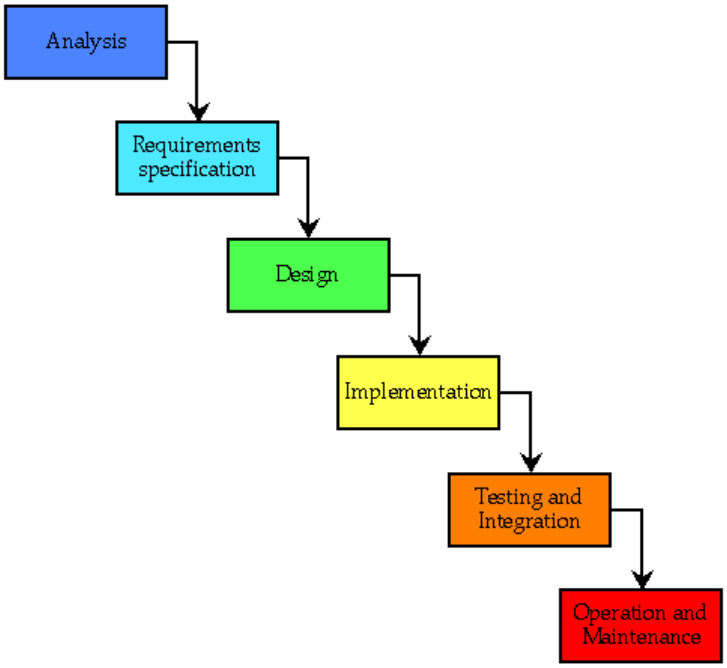
INTRODUCTION

Politeknik Balik Pulau Sport Center (PBUSCAPP) is a mobile application that uses Android software. Various functions are available in this application. Among the user must fill in a username and password to use this application fully. This application can run on all Android software latest version 4.0 until 6.0. Users can store information storage aout sport items using only mobile phones.

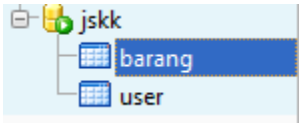
SDLC

System development lifecycle (SDLC) – the entire process consisting of all activities required to build, launch, and maintain an information system

- Identify the problem or need and obtain approval
- Plan and monitor the project
- Discover and understand the details of the problem or need
- Design the system components that solve the problem or satisfy the need
- Build, test, and integrate system components
- Complete system tests and then deploy the solution



Waterfall Model



Database for this Project

PROBLEM STATEMENT

Department of Sports, Curriculum and Culture (JSKK) Polytechnics Balik Pulau have problems to keep records of data on sports goods in the sports room. They used manually for storing data in a log book. It is one of the ways that are not effective. So we've built a mobile application that is capable of storing all the information regarding the storage of sporting goods stored in the database.

OBJECTIVE PROJECT

Among the objectives of this project are:

1. The user can store data via an easy alternative way without having to use manual.
2. The user can insert, delete and update any data they store in the database.
3. This application has a security feature where users can only access this application by entering your username and password.
4. To save time staff sports center without having to write by hand
5. Consumers easier to use this Application
6. Do not need to worry when the loss of data of the Sport Center.

CODES

Main

```
#Region  Project Attributes
	#ApplicationLabel: PBUSCAPP
	#VersionCode: 14
	#VersionName: 1.4
	#VersionName: Beta
	'SupportedOrientations possible values: unspecified, landscape or portrait.
	#SupportedOrientations: portrait
	#CanInstallToExternalStorage: False
#End Region

#Region  Activity Attributes
	#FullScreen: False
	#IncludeTitle: False
#End Region

Sub Process_Globals
	'These global variables will be declared once when the application starts.
	'These variables can be accessed from all modules.
Dim timer1 As Timer
Public x1, x2, x3 As String

End Sub

Sub Globals
	'These global variables will be redeclared each time the activity is created.
	'These variables can only be accessed from this module.



	Dim ProgressBar1 As ProgressBar
	Dim num As Int


	Private Button1 As Button
	Private btnScramble As Button
	Private btnAbout As Button

	Private btnVolume As Button
	Private IRCircularRing1 As IRCircularRing
End Sub

Sub Activity_Create(FirstTime As Boolean)
```

```
        'Do not forget to load the layout file created with the visual designer. For example:
        Activity.LoadLayout("intro")

IRCircularRing1.startAnim
                timer1.Initialize("timer1",50)
        timer1.Enabled = True
        ProgressBar1.Top = 94%y
        ProgressBar1.Left  = 0%x
        ProgressBar1.Width = 100%x
End Sub

Sub Activity_Resume

End Sub

Sub Activity_Pause (UserClosed As Boolean)

End Sub

Sub timer1_tick
num = num +1
ProgressBar1.Progress = num

If ProgressBar1.Progress == 2 Then

End If

If ProgressBar1.Progress == 70 Then


End If


If ProgressBar1.Progress == 100 Then
timer1.Enabled = False
IRCircularRing1.stopAnim
StartActivity("Login")
Activity.Finish
End If
End Sub
```

Login

```
#Region  Activity Attributes
        #FullScreen: False
        #IncludeTitle: True
#End Region
#Extends: android.support.v7.app.AppCompatActivity
Sub Process_Globals
```

```
	Dim SQL As SQL

	Dim DBDir As String : DBDir = File.DirDefaultExternal

	Dim DBName As String : DBName = "jskk.db"

	Dim UserTable As String = "user"
	Dim BarangTable As String = "barang"
	Public intro,intro2 As MediaPlayer
End Sub

Sub Globals
	'These global variables will be redeclared each time the activity is created.
	'These variables can only be accessed from this module.

	Private EdtUsername As EditText
	Private EdtPassword As EditText
	Private BtnLogin As Button

		Dim AC As AppCompat
	Dim ABHelper As ACActionBar
	Private pContent As Panel
Private ActionBar As ACToolBarLight


End Sub

Sub Activity_Create(FirstTime As Boolean)
	'Do not forget to load the layout file created with the visual designer. For example:


	Activity.LoadLayout("main")
	pContent.LoadLayout("login")
	ActionBar.Title = "LOGIN"
	ActionBar.SubTitle = ""

	intro.Initialize2("intro")
	intro.Load(File.DirAssets, "enterauthorizationcode.mp3")
	intro.Play

	intro2.Initialize2("intro2")
	intro2.Load(File.DirAssets, "access_granted.mp3")

	Activity.AddMenuItem("Exit", "Menu")
	EdtUsername.RequestFocus

	If FirstTime Then
```

```
		If File.Exists(DBDir, DBName) = False Then
			File.Copy(File.DirAssets, DBName, DBDir, DBName)
		End If


		SQL.Initialize(DBDir, DBName, True)
	End If
End Sub

Sub Activity_Resume
	EdtUsername.RequestFocus
	EdtUsername.Text=""
	EdtPassword.Text=""

End Sub

Sub Activity_Pause (UserClosed As Boolean)

End Sub

Sub BtnLogin_Click

	If EdtUsername.Text = "" Then



		Msgbox("Please enter Username", "Warning")
		Return
	End If

	If EdtPassword.Text = "" Then
		Msgbox("Please enter Password", "Warning")
		Return
	End If

	Dim m As Map = CheckLogin(EdtUsername.Text, EdtPassword.Text)


	If m.IsInitialized = True Then
		intro2.Play
		Msgbox("Hye, " & m.Get("username") & CRLF & _
		"Welcome to PBUSCAPP", "SUCCESS")
		StartActivity(Home)
	Else
		Msgbox("Username @ Password is wrong", "Failed Login")
	End If
End Sub
```

```
Sub CheckLogin(Username As String, Password As String) As Map
	Dim Query As String
	Query = "select * from " & UserTable & " where username = ? and password = ?"
	Dim M As Map = DbUtils.ExecuteMap(SQL, Query, Array As String(Username, Password))
	Return M
End Sub

Sub Menu_Click()
Dim bmp As Bitmap
Dim choice As Int
bmp.Initialize(File.DirAssets, "help.png")
choice = Msgbox2(" Quit now ?", "Comfirmation ", "Yes", "", "No", bmp)
If choice = DialogResponse.POSITIVE Then
ExitApplication

End If
End Sub
```

Home

```
#Region Module Attributes
	#FullScreen: False
	#IncludeTitle: True
#End Region
#Extends: android.support.v7.app.AppCompatActivity

Sub Process_Globals

Public intro,intro2 As MediaPlayer

End Sub

Sub Globals


		Dim AC As AppCompat
	Dim ABHelper As ACActionBar
	Private pContent As Panel
Private ActionBar As ACToolBarLight
Dim BtnBarang As Button

	Private btnLogOut As Button
	Private btnImage As Button
	Private btnAbout As Button
	Private btnItem As Button
End Sub

Sub Activity_Create(FirstTime As Boolean)



	Activity.LoadLayout("main")
	pContent.LoadLayout("menu")
	ActionBar.Title = "MENU"
	ActionBar.SubTitle = ""
	Activity.AddMenuItem("About Me", "Menu")

	intro.Initialize2("intro")
	intro.Load(File.DirAssets, "wlcome.mp3")
	intro.Play

	intro2.Initialize2("intro2")
	intro2.Load(File.DirAssets, "enterauthorizationcode.mp3")
End Sub
```

```
Sub Activity_Resume

End Sub

Sub Activity_Pause (UserClosed As Boolean)

End Sub


Sub Menu_Click()
Msgbox("PBUSCAPP Develop By Nazrul Wazir @ 2016", "About Me")
End Sub

Sub btnLogOut_Click
        Dim bmp As Bitmap
Dim choice As Int
bmp.Initialize(File.DirAssets, "help.png")
choice = Msgbox2(" Log Out now ?", "Comfirmation ", "Yes", "", "No", bmp)
If choice = DialogResponse.POSITIVE Then
                intro2.Play
Activity.Finish
End If
End Sub

Sub btnImage_Click
        StartActivity(image)
End Sub

Sub btnAbout_Click
        StartActivity(about)
End Sub

Sub btnItem_Click
        StartActivity(InventoryList)
End Sub

Sub Activity_KeyPress (KeyCode As Int) As Boolean
 If KeyCode = KeyCodes.KEYCODE_BACK  Then

   Select Msgbox2("Log Out Now?", "", "OK", "","Cancel", Null)
   Case DialogResponse.NEGATIVE
    Return True
   Case DialogResponse.CANCEL
    Return True
   Case DialogResponse.POSITIVE
                intro2.Play
         Msgbox("Thank You!","")
```

```
        Activity.Finish
  End Select

End If
End Sub
```

DBUtils (Class Module)

```
' Code module
' Version 1.09
'
' Modified: 15.06.2011 Markus Stipp
'		- Version control of database (GetDBVersion and SetDBVersion Subs)
'		- renamed cursor variables from c to cur since I often use a module named C for constants.
'
' Modified: 30.03.2012 Klaus Christl (klaus)
' - Added DeletRecord		deletes a record, code from Erel
' - Added UpdateRecord2 updates more than one field in a record, code from vasper

Sub Process_Globals
	Dim DB_REAL, DB_INTEGER, DB_BLOB, DB_TEXT As String
	DB_REAL = "REAL"
	DB_INTEGER = "INTEGER"
	DB_BLOB = "BLOB"
	DB_TEXT = "TEXT"
	Dim HtmlCSS As String
	HtmlCSS = "table {width: 100%;border: 1px solid #cef;text-align: left; }" _
		& " th { font-weight: bold;		background-color: #acf; border-bottom: 1px solid
#cef; }" _
		& "td,th {		padding: 4px 5px; }" _
		& ".odd {background-color: #def; } .odd td {border-bottom: 1px solid #cef; }" _
		& "a { text-decoration:none; color: #000;}"
End Sub

'Copies a database file that was added in the Files tab. The database must be copied to a writable
location.
'This method copies the database to the storage card. If the storage card is not available the file is
copied to the internal folder.
'The target folder is returned.
'If the database file already exists then no copying is done.
Sub CopyDBFromAssets (FileName As String) As String
	Dim TargetDir As String
	If File.ExternalWritable Then TargetDir = File.DirDefaultExternal Else TargetDir =
File.DirInternal
	If File.Exists(TargetDir, FileName) = False Then
		File.Copy(File.DirAssets, FileName, TargetDir, FileName)
	End If
	Return TargetDir
End Sub

'Creates a new table with the given name.
'FieldsAndTypes - A map with the fields names as keys and the types as values.
'You can use the DB_... constants for the types.
'PrimaryKey - The column that will be the primary key. Pass empty string if not needed.
```

```
Sub CreateTable(SQL As SQL, TableName As String, FieldsAndTypes As Map, PrimaryKey As String)
	Dim sb As StringBuilder
	sb.Initialize
	sb.Append("(")
	For i = 0 To FieldsAndTypes.Size - 1
		Dim field, ftype As String
		field = FieldsAndTypes.GetKeyAt(i)
		ftype = FieldsAndTypes.GetValueAt(i)
		If i > 0 Then sb.Append(", ")
		sb.Append("[").Append(field).Append("] ").Append(ftype)
		If field = PrimaryKey Then sb.Append(" PRIMARY KEY")
	Next
	sb.Append(")")
	Dim query As String
	query = "CREATE TABLE IF NOT EXISTS [" & TableName & "] " & sb.ToString
	Log("CreateTable: " & query)
	SQL.ExecNonQuery(query)
End Sub

'Deletes the given table.
Sub DropTable(SQL As SQL, TableName As String)
	Dim query As String
	query = "DROP TABLE IF EXISTS [" & TableName & "]"
	Log("DropTable: " & query)
	SQL.ExecNonQuery(query)
End Sub

'Inserts the data to the table.
'ListOfMaps - A list with maps as items. Each map represents a record where the map keys are the
columns names
'and the maps values are the values.
'Note that you should create a new map for each record (this can be done by calling Dim to redim the
map).
Sub InsertMaps(SQL As SQL, TableName As String, ListOfMaps As List)
	Dim sb, columns, values As StringBuilder
	'Small check for a common error where the same map is used in a loop
	If ListOfMaps.Size > 1 And ListOfMaps.Get(0) = ListOfMaps.Get(1) Then
		Log("Same Map found twice in list. Each item in the list should include a different
map object.")
		ToastMessageShow("Same Map found twice in list. Each item in the list should
include a different map object.", True)
		Return
	End If
	SQL.BeginTransaction
	Try
		For i1 = 0 To ListOfMaps.Size - 1
			sb.Initialize
			columns.Initialize
```

```
                        values.Initialize
                        Dim listOfValues As List
                        listOfValues.Initialize
                        sb.Append("INSERT INTO [" & TableName & "] (")
                        Dim m As Map
                        m = ListOfMaps.Get(i1)
                        For i2 = 0 To m.Size - 1
                                Dim col As String
                                Dim value As Object
                                col = m.GetKeyAt(i2)
                                value = m.GetValueAt(i2)
                                If i2 > 0 Then
                                        columns.Append(", ")
                                        values.Append(", ")
                                End If
                                columns.Append("[").Append(col).Append("]")
                                values.Append("?")
                                listOfValues.Add(value)
                        Next
                        sb.Append(columns.ToString).Append(") VALUES
(").Append(values.ToString).Append(")")
                        If i1 = 0 Then Log("InsertMaps (first query out of " & ListOfMaps.Size & "): " &
sb.ToString)
                        SQL.ExecNonQuery2(sb.ToString, listOfValues)
                Next
                SQL.TransactionSuccessful
        Catch
                ToastMessageShow(LastException.Message, True)
                Log(LastException)
        End Try
        SQL.EndTransaction
End Sub

' updates a single field in a record
' Field is the column name
Sub UpdateRecord(SQL As SQL, TableName As String, Field As String, NewValue As Object, _
        WhereFieldEquals As Map)
        Dim sb As StringBuilder
        sb.Initialize
        sb.Append("UPDATE [").Append(TableName).Append("] SET [").Append(Field).Append("] = ?
WHERE ")
        If WhereFieldEquals.Size = 0 Then
                Log("WhereFieldEquals map empty!")
                Return
        End If
        Dim args As List
        args.Initialize
        args.Add(NewValue)
```

```
	For i = 0 To WhereFieldEquals.Size - 1
		If i > 0 Then sb.Append(" AND ")
		sb.Append("[").Append(WhereFieldEquals.GetKeyAt(i)).Append("] = ?")
		args.Add(WhereFieldEquals.GetValueAt(i))
	Next
	Log("UpdateRecord: " & sb.ToString)
	SQL.ExecNonQuery2(sb.ToString, args)
End Sub

' updates multiple fields in a record
' in the Fields map the keys are the column names
Sub UpdateRecord2(SQL As SQL, TableName As String, Fields As Map, WhereFieldEquals As Map)
	If WhereFieldEquals.Size = 0 Then
		Log("WhereFieldEquals map empty!")
		Return
	End If
	If Fields.Size = 0 Then
		Log("Fields empty")
		Return
	End If
	Dim sb As StringBuilder
	sb.Initialize
	sb.Append("UPDATE [").Append(TableName).Append("] SET ")
	Dim args As List
	args.Initialize
	For i=0 To Fields.Size-1
		If i<>Fields.Size-1 Then
			sb.Append("[").Append(Fields.GetKeyAt(i)).Append("]=?,")
		Else
			sb.Append("[").Append(Fields.GetKeyAt(i)).Append("]=?")
		End If
		args.Add(Fields.GetValueAt(i))
	Next

	sb.Append(" WHERE ")
	For i = 0 To WhereFieldEquals.Size - 1
		If i > 0 Then
			sb.Append(" AND ")
		End If
		sb.Append("[").Append(WhereFieldEquals.GetKeyAt(i)).Append("] = ?")
		args.Add(WhereFieldEquals.GetValueAt(i))
	Next
	Log("UpdateRecord: " & sb.ToString)
	SQL.ExecNonQuery2(sb.ToString, args)
End Sub

'Executes the query and returns the result as a list of arrays.
'Each item in the list is a strings array.
```

```
'StringArgs - Values to replace question marks in the query. Pass Null if not needed.
'Limit - Limits the results. Pass 0 for all results.
Sub ExecuteMemoryTable(SQL As SQL, Query As String, StringArgs() As String, Limit As Int) As List
	Dim cur As Cursor
	If StringArgs <> Null Then
		cur = SQL.ExecQuery2(Query, StringArgs)
	Else
		cur = SQL.ExecQuery(Query)
	End If
	Log("ExecuteMemoryTable: " & Query)
	Dim table As List
	table.Initialize
	If Limit > 0 Then Limit = Min(Limit, cur.RowCount) Else Limit = cur.RowCount
	For row = 0 To Limit - 1
		cur.Position = row
		Dim values(cur.ColumnCount) As String
		For col = 0 To cur.ColumnCount - 1
			values(col) = cur.GetString2(col)
		Next
		table.Add(values)
	Next
	cur.Close
	Return table
End Sub

'Executes the query and returns a Map with the column names as the keys
'and the first record values As the entries values.
'The keys are lower cased.
'Returns Null if no results found.
Sub ExecuteMap(SQL As SQL, Query As String, StringArgs() As String) As Map
	Dim cur As Cursor
	If StringArgs <> Null Then
		cur = SQL.ExecQuery2(Query, StringArgs)
	Else
		cur = SQL.ExecQuery(Query)
	End If
	Log("ExecuteMap: " & Query)
	If cur.RowCount = 0 Then
		Log("No records found.")
		Return Null
	End If
	Dim res As Map
	res.Initialize
	cur.Position = 0
	For i = 0 To cur.ColumnCount - 1
		res.Put(cur.GetColumnName(i).ToLowerCase, cur.GetString2(i))
	Next
	cur.Close
```

```
	Return res
End Sub

'Executes the query and fills the Spinner with the values in the first column
Sub ExecuteSpinner(SQL As SQL, Query As String, StringArgs() As String, Limit As Int, Spinner1 As
Spinner)
	Spinner1.Clear
	Dim Table As List
	Table = ExecuteMemoryTable(SQL, Query, StringArgs, Limit)
	Dim Cols() As String
	For i = 0 To Table.Size - 1
		Cols = Table.Get(i)
		Spinner1.Add(Cols(0))
	Next
End Sub

'Executes the query and fills the ListView with the value.
'If TwoLines is true then the first column is mapped to the first line and the second column is mapped
'to the second line.
'In both cases the value set to the row is the array with all the records values.
Sub ExecuteListView(SQL As SQL, Query As String, StringArgs() As String, Limit As Int, ListView1 As
ListView, _
	TwoLines As Boolean)
	ListView1.Clear
	Dim Table As List
	Table = ExecuteMemoryTable(SQL, Query, StringArgs, Limit)
	Dim Cols() As String
	For i = 0 To Table.Size - 1
		Cols = Table.Get(i)
		If TwoLines Then
			ListView1.AddTwoLines2(Cols(0), Cols(1), Cols)
		Else
			ListView1.AddSingleLine2(Cols(0), Cols)
		End If
	Next
End Sub

'Executes the given query and creates a Map that you can pass to JSONGenerator and generate JSON
text.
'DBTypes - Lists the type of each column in the result set.
'Usage example: (don't forget to add a reference to the JSON library)
'	Dim gen As JSONGenerator
'	gen.Initialize(DBUtils.ExecuteJSON(SQL, "SELECT Id, Birthday FROM Students", Null, _
'		0, Array As String(DBUtils.DB_TEXT, DBUtils.DB_INTEGER)))
'	Dim JSONString As String
'	JSONString = gen.ToPrettyString(4)
'	Msgbox(JSONString, "")
```

```
Sub ExecuteJSON (SQL As SQL, Query As String, StringArgs() As String, Limit As Int, DBTypes As List) As
Map
	Dim table As List
	Dim cur As Cursor
	If StringArgs <> Null Then
		cur = SQL.ExecQuery2(Query, StringArgs)
	Else
		cur = SQL.ExecQuery(Query)
	End If
	Log("ExecuteJSON: " & Query)
	Dim table As List
	table.Initialize
	If Limit > 0 Then Limit = Min(Limit, cur.RowCount) Else Limit = cur.RowCount
	For row = 0 To Limit - 1
		cur.Position = row
		Dim m As Map
		m.Initialize
		For i = 0 To cur.ColumnCount - 1
			Select DBTypes.Get(i)
				Case DB_TEXT
					m.Put(cur.GetColumnName(i), cur.GetString2(i))
				Case DB_INTEGER
					m.Put(cur.GetColumnName(i), cur.GetLong2(i))
				Case DB_REAL
					m.Put(cur.GetColumnName(i), cur.GetDouble2(i))
				Case Else
					Log("Invalid type: " & DBTypes.Get(i))
			End Select
		Next
		table.Add(m)
	Next
	cur.Close
	Dim root As Map
	root.Initialize
	root.Put("root", table)
	Return root
End Sub

'Creates a html text that displays the data in a table.
'The style of the table can be changed by modifying HtmlCSS variable.
Sub ExecuteHtml(SQL As SQL, Query As String, StringArgs() As String, Limit As Int, Clickable As
Boolean) As String
	Dim Table As List
	Dim cur As Cursor
	If StringArgs <> Null Then
		cur = SQL.ExecQuery2(Query, StringArgs)
	Else
		cur = SQL.ExecQuery(Query)
```

```
	End If
	Log("ExecuteHtml: " & Query)
	If Limit > 0 Then Limit = Min(Limit, cur.RowCount) Else Limit = cur.RowCount
	Dim sb As StringBuilder
	sb.Initialize
	sb.Append("<html><body>").Append(CRLF)
	sb.Append("<style type='text/css'>").Append(HtmlCSS).Append("</style>").Append(CRLF)
	sb.Append("<table><tr>").Append(CRLF)
	For i = 0 To cur.ColumnCount - 1
		sb.Append("<th>").Append(cur.GetColumnName(i)).Append("</th>")
	Next

'	For i = 0 To cur.ColumnCount - 1
'		If i = 1 Then
'			sb.Append("<th
style='width:200px;'>").Append(cur.GetColumnName(i)).Append("</th>")
'		Else
'			sb.Append("<th>").Append(cur.GetColumnName(i)).Append("</th>")
'		End If
'	Next

	sb.Append("</tr>").Append(CRLF)
	For row = 0 To Limit - 1
		cur.Position = row
		If row Mod 2 = 0 Then
			sb.Append("<tr>")
		Else
			sb.Append("<tr class='odd'>")
		End If
		For i = 0 To cur.ColumnCount - 1
			sb.Append("<td>")
			If Clickable Then
				sb.Append("<a href='http://").Append(i).Append(".")
				sb.Append(row)
				sb.Append(".com'>").Append(cur.GetString2(i)).Append("</a>")
			Else
				sb.Append(cur.GetString2(i))
			End If
			sb.Append("</td>")
		Next
		sb.Append("</tr>").Append(CRLF)
	Next
	cur.Close
	sb.Append("</table></body></html>")
	Return sb.ToString
End Sub
```

```
'Gets the current version of the database. If the DBVersion table does not exist it is created and the
current
'version is set to version 1.
Sub GetDBVersion (SQL As SQL) As Int
	Dim count, version As Int
	count = SQL.ExecQuerySingleResult("SELECT count(*) FROM sqlite_master WHERE
Type='table' AND name='DBVersion'")
	If count > 0 Then
		version = SQL.ExecQuerySingleResult("SELECT version FROM DBVersion")
	Else
		'Create the versions table.
		Dim m As Map
		m.Initialize
		m.Put("version", DB_INTEGER)
		CreateTable(SQL, "DBVersion", m, "version")

		SQL.ExecNonQuery("INSERT INTO DBVersion VALUES (1)")

		version = 1
	End If

	Return version
End Sub

'Sets the database version to the given version number.
Sub SetDBVersion (SQL As SQL, Version As Int)
	SQL.ExecNonQuery2("UPDATE DBVersion set version = ?", Array As Object(Version))
End Sub

' deletes a record
Sub DeleteRecord(SQL As SQL, TableName As String, WhereFieldEquals As Map)
  Dim sb As StringBuilder
  sb.Initialize
  sb.Append("DELETE FROM [").Append(TableName).Append("] WHERE ")
  If WhereFieldEquals.Size = 0 Then
    Log("WhereFieldEquals map empty!")
    Return
  End If
  Dim args As List
  args.Initialize
  For i = 0 To WhereFieldEquals.Size - 1
    If i > 0 Then sb.Append(" AND ")
    sb.Append("[").Append(WhereFieldEquals.GetKeyAt(i)).Append("] = ?")
    args.Add(WhereFieldEquals.GetValueAt(i))
  Next
  Log("DeleteRecord: " & sb.ToString)
  SQL.ExecNonQuery2(sb.ToString, args)
End Sub
```

InventoryList

```
#Region  Activity Attributes
	#FullScreen: False
	#IncludeTitle: True
#End Region
#Extends: android.support.v7.app.AppCompatActivity
Sub Process_Globals
	'These global variables will be declared once when the application starts.
	'These variables can be accessed from all modules.

End Sub

Sub Globals

	Dim LvBarang As ListView

	Dim BtnCreate As Button

	Dim AC As AppCompat
	Dim ABHelper As ACActionBar
	Private pContent As Panel
Private ActionBar As ACToolBarLight
End Sub

Sub Activity_Create(FirstTime As Boolean)


		Activity.LoadLayout("main")
pContent.LoadLayout("items")
	ActionBar.Title = "ITEMS JSKK"
	ActionBar.SubTitle = ""
		ABHelper.Initialize
ABHelper.ShowUpIndicator = True
ActionBar.InitMenuListener



	LvBarang.Initialize("LvBarang")

	BtnCreate.Initialize("BtnCreate")

	BtnCreate.Text = "Create Item"

	pContent.AddView(LvBarang, 0,0,100%x,85%y)
```

```
End Sub

Sub Activity_Resume
	'Refresh data
	FillLvBarang
End Sub

Sub Activity_Pause (UserClosed As Boolean)

End Sub

Sub FillLvBarang

	query = " SELECT kode, nama, id FROM " & Login.BarangTable
	DbUtils.ExecuteListView(Login.SQL, query, Null, 0, LvBarang, True)
End Sub

Sub BtnCreate_Click

	InventoryView.ID = -1
	StartActivity(InventoryView)
End Sub

Sub LvBarang_ItemClick (Position As Int, Value As Object)
	Dim v(2) As String = Value

	InventoryView.ID = v(2)
	StartActivity(InventoryView)
End Sub

Sub ActionBar_NavigationItemClick
  Activity.Finish
End Sub
```

InventoryView

```
#Region  Activity Attributes
	#FullScreen: False
	#IncludeTitle: True
#End Region
#Extends: android.support.v7.app.AppCompatActivity

Sub Process_Globals
	'These global variables will be declared once when the application starts.
	'These variables can be accessed from all modules.


	Dim ID As Int
End Sub

Sub Globals
	'These global variables will be redeclared each time the activity is created.
	'These variables can only be accessed from this module.

	Dim PnlBarang As Panel

	Dim LblKode As Label
	Dim LblNama As Label
	Dim LblKeterangan As Label
	Dim Lblkuantiti As Label
	Dim Lblstatus As Label

	Dim EdtKode As EditText
	Dim EdtNama As EditText
	Dim EdtKeterangan As EditText
	Dim EdtKuantiti As EditText
	Dim Edtstatus As EditText

	Dim BtnSave As Button
	Dim BtnDelete As Button

	Dim AC As AppCompat
	Dim ABHelper As ACActionBar
	Private pContent As Panel
Private ActionBar As ACToolBarLight
End Sub

Sub Activity_Create(FirstTime As Boolean)
	'Do not forget to load the layout file created with the visual designer. For example:
	'Activity.LoadLayout("Layout1")

		Activity.LoadLayout("main")
```

```
pContent.LoadLayout("create_items")
	ActionBar.Title = "CREATE ITEMS JSKK"
	ActionBar.SubTitle = ""
		ABHelper.Initialize
ABHelper.ShowUpIndicator = True
ActionBar.InitMenuListener



	InitObject

	SetLabel

	LoadBarang

	SetObjectView
End Sub

Sub Activity_Resume

End Sub

Sub Activity_Pause (UserClosed As Boolean)

End Sub

Sub InitObject
	LblKode.Initialize("")
	LblNama.Initialize("")
	LblKeterangan.Initialize("")
	Lblkuantiti.Initialize("")
	Lblstatus.Initialize("")

	EdtKode.Initialize("EdtKode")
	EdtNama.Initialize("EdtNama")
	EdtKeterangan.Initialize("EdtKeterangan")
	EdtKuantiti.Initialize("EdtKuantiti")
	Edtstatus.Initialize("EdtKuantiti")

	BtnSave.Initialize("BtnSave")
	BtnDelete.Initialize("BtnDelete")

	PnlBarang.Initialize("PnlBarang")
End Sub

Sub SetLabel
	LblKode.Text = "Code"
	LblNama.Text = "Name Items"
```

```
	LblKeterangan.Text = "Details Items"
	Lblkuantiti.Text = "Quantity Items"
	Lblstatus.Text = "Status Items"


	EdtKode.Enabled = False


	If ID = -1 Then
		BtnSave.Text = "Save"
		BtnDelete.Enabled = False
	Else
		BtnSave.Text = "Update"
		BtnDelete.Text = True
	End If
	BtnDelete.Text = "Delete"
End Sub

Sub LoadBarang

	Dim query As String
	query = " SELECT * FROM " & Login.BarangTable & " WHERE id = ?"

	Dim m As Map
	m = DbUtils.ExecuteMap(Login.SQL, query, Array As String(ID))

	If m.IsInitialized = False Then
		If ID = -1 Then
			Dim kode As String = GenerateKode
			EdtKode.Text = kode
			EdtNama.Text = ""
			EdtKeterangan.Text = ""
			EdtKuantiti.Text = ""
			Edtstatus.Text = ""
		End If

	Else
		If m.Get("kode") <> Null Then
		  EdtKode.Text = m.Get("kode")
		End If

		If m.Get("nama") <> Null Then
			Activity.Title = "View: " & m.Get("nama")
		  EdtNama.Text = m.Get("nama")
		End If

		If m.Get("keterangan") <> Null Then
		  EdtKeterangan.Text = m.Get("keterangan")
```

```
                End If

                If m.Get("kuantiti") <> Null Then
                   EdtKuantiti.Text = m.Get("kuantiti")
                End If

                If m.Get("status") <> Null Then
                   Edtstatus.Text = m.Get("status")
                End If
        End If
End Sub

Sub SetObjectView
        Dim ctop As Int = 20dip
        Dim labelHeight As Int = 30dip
        Dim textHeight As Int = 40dip


        PnlBarang.AddView(LblKode, 20dip, ctop, 100%x-40dip, 30dip) : ctop = ctop + labelHeight
        PnlBarang.AddView(EdtKode, 20dip, ctop, 100%x-40dip, 40dip) : ctop = ctop + textHeight


        PnlBarang.AddView(LblNama, 20dip, ctop, 100%x-40dip, 30dip) : ctop = ctop + labelHeight
        PnlBarang.AddView(EdtNama, 20dip, ctop, 100%x-40dip, 40dip) : ctop = ctop + textHeight

        PnlBarang.AddView(Lblkuantiti, 20dip, ctop, 100%x-40dip, 30dip) : ctop = ctop + labelHeight
        PnlBarang.AddView(EdtKuantiti, 20dip, ctop, 100%x-40dip, 40dip) : ctop = ctop + textHeight

        PnlBarang.AddView(LblKeterangan, 20dip, ctop, 100%x-40dip, 30dip) : ctop = ctop +
labelHeight
        PnlBarang.AddView(EdtKeterangan, 20dip, ctop, 100%x-40dip, 40dip) : ctop = ctop +
textHeight

        PnlBarang.AddView(Lblstatus, 20dip, ctop, 100%x-40dip, 30dip) : ctop = ctop + labelHeight
        PnlBarang.AddView(Edtstatus, 20dip, ctop, 100%x-40dip, 40dip) : ctop = ctop + textHeight


        Activity.AddView(PnlBarang, 0,0,100%x, 85%y)
        Activity.AddView(BtnSave, 0, 85%y, 50%x, 15%y)
        Activity.AddView(BtnDelete, 50%x, 85%y, 50%x, 15%y)
End Sub

Sub GenerateKode
        Dim q As String = "select id from " & Login.BarangTable
        Dim qs As List = DbUtils.ExecuteMemoryTable(Login.SQL, q, Null, 0)

        Dim count As String = qs.Size + 1
        If count < 10 Then
```

```
                count = "000" & count
        Else If count < 100 Then
                count = "00" & count
        Else If count < 1000 Then
                count = "000" & count
        Else
                count = count
        End If

        Log("count: " & count)

        Dim kd As String = "ITEM-" & count

        Return kd
End Sub

Sub BtnSave_Click()
        Dim listOfMaps As List : listOfMaps.Initialize
        Dim m As Map : m.Initialize
        m.put("kode", EdtKode.Text)
        m.put("nama", EdtNama.Text)
        m.put("keterangan", EdtKeterangan.Text)
        m.put("kuantiti", EdtKuantiti.Text)
        m.put("status", Edtstatus.Text)
        listOfMaps.Add(m)


        If ID = -1 Then
                DbUtils.InsertMaps(Login.SQL, Login.BarangTable, listOfMaps)
                ToastMessageShow("Barang has been created.", True)

                Activity.Finish
        Else
                Dim w As Map : w.Initialize
                w.Put("id", ID)
                DbUtils.UpdateRecord2(Login.SQL, Login.BarangTable, m, w)
                ToastMessageShow("Barang has been updated.", True)

                LoadBarang
                Activity.Finish
        End If
End Sub

Sub BtnDelete_Click()
        Dim result As Int = Msgbox2( "Delete : " & EdtNama.Text & "?", "Confirmation", "Yes", "No",
"", _
           LoadBitmap (File.DirAssets, "confirm.png"))
        If result = DialogResponse.Positive Then
```

```
                Dim w As Map : w.Initialize
                w.Put("id", ID)
                DbUtils.DeleteRecord(Login.SQL, Login.BarangTable , w)
                ToastMessageShow("Item has been deleted.", True)
                Activity.Finish
        End If
End Sub

Sub ActionBar_NavigationItemClick
  Activity.Finish
End Sub
```

SlidingPanels (Class Module)

```
'Code module
Sub Process_Globals
	Type SlidingData (firstTime As Boolean, currentPanel As Int, Panels() As Panel,
LeftAnimations() As Animation, RightAnimations() As Animation, targetPanel As Int)
End Sub

Sub Initialize (sd As SlidingData, SlidingDuration As Int)
	duration = SlidingDuration
'	Home.tmrAnimation.Initialize("tmrAnimation", 2)
	Dim a(2) As Animation
	sd.LeftAnimations = a
	Dim a(2) As Animation
	sd.RightAnimations = a
	'Initialize the animation objects. We need two objects for each direction as both the current
panel and the new panel are animated.
	For i = 0 To 1
		sd.leftAnimations(i).InitializeTranslate("animation" & i, 0, 0, -100%x, 0)
		sd.leftAnimations(i).Duration = SlidingDuration
		sd.rightAnimations(i).InitializeTranslate("animation" & i, 0, 0, 100%x, 0)
		sd.rightAnimations(i).Duration = SlidingDuration
	Next
	For i = 0 To sd.Panels.Length - 1
		sd.Panels(i).Left = 100%x 'Move the panels right of the screen
	Next
	sd.firstTime = True
End Sub

Sub ChangePanel(sd As SlidingData, left As Boolean)
	If left Then
		If sd.firstTime = False Then 'remove current panel if such exists (it will not be the case
on the first call).
			sd.leftAnimations(0).Start(sd.panels(sd.currentPanel)) 'Animate current panel
and move it out
		Else
			sd.firstTime = False
		End If
		sd.leftAnimations(1).Start(sd.panels((sd.currentPanel + 1) Mod sd.Panels.Length))
'Animate new panel
		sd.currentPanel = (sd.currentPanel + 1) Mod sd.Panels.Length
	Else
		Dim leftPanel As Int
		leftPanel = (sd.currentPanel + sd.Panels.Length - 1) Mod sd.Panels.Length
		sd.panels(leftPanel).left = -100%x
		sd.rightAnimations(0).Start(sd.panels(sd.currentPanel))
		sd.rightAnimations(1).Start(sd.panels(leftPanel))
		sd.currentPanel = leftPanel
```

```
        End If
End Sub
Sub AnimationEnd (sd As SlidingData)
        sd.panels(sd.currentPanel).Left = 0 'Set the position of the new panel
        For i = 0 To sd.panels.Length - 1
                If i <> sd.currentPanel Then sd.panels(i).Left = 100%x 'Move all other panels right of
the screen.
        Next
End Sub
```

Image

```
#Region Module Attributes
        #FullScreen: False
        #IncludeTitle: True
#End Region
#Extends: android.support.v7.app.AppCompatActivity

Sub Process_Globals
Dim tmrAnimation As Timer
Dim currentPanelBeforePaused As Int
Dim tmrSlider As Timer

End Sub

Sub Globals
        Dim bmp0, bmp1, bmp2, bmp3 As BitmapDrawable
   Dim sd As SlidingData
        Dim startX, startY As Float
        Dim SlidingDuration As Int
        SlidingDuration = 700
        Dim offsetX As Int = 45%x
        Dim imgs(5) As ImageView
        Dim cd , cd2 As ColorDrawable
        Dim  introwel As MediaPlayer
        Private btnKata As Button


        Private btnVideo As Button
                Dim AC As AppCompat
        Dim ABHelper As ACActionBar
        Private pContent As Panel
Private ActionBar As ACToolBarLight


End Sub
```

```
Sub Activity_Create(FirstTime As Boolean)



	Activity.LoadLayout("main")
	pContent.LoadLayout("image")
	ActionBar.Title = "IMAGE PBU"
	ActionBar.SubTitle = ""

	ABHelper.Initialize
ABHelper.ShowUpIndicator = True
ActionBar.InitMenuListener



	Dim panels(5) As Panel
	cd.Initialize(Colors.black,10dip)
	cd2.Initialize(Colors.DarkGray,10dip)
	For i = 0 To 4
	 panels(i).Initialize("panels")
	 imgs(i).Initialize("imgs")
	 pContent.Color=Colors.White
	 pContent.AddView(panels(i),0%x,0%y,100%x,88%y)

panels(i).SetBackgroundImage(LoadBitmapSample(File.DirAssets,(i+1)&".jpg",panels(i).Width,panels(i
).Height))
	 pContent.AddView(imgs(i),offsetX,(panels(i).Top + panels(i).Height) + 5dip,10dip,10dip)
	 If i = 0 Then
		imgs(i).Background = cd2
	 Else
		imgs(i).Background = cd
	 End If
	 offsetX = offsetX + 10dip
	Next
	sd.Initialize
	sd.panels = panels
	SlidingPanels.Initialize(sd, SlidingDuration)
	sd.targetPanel = -1
	sd.currentPanel = currentPanelBeforePaused - 1
	ChangePanel(True)
	tmrSlider.Initialize("tmrSlider",5000)
	tmrSlider.Enabled = True


End Sub

Sub tmrSlider_Tick
```

```
 ChangePanel(True)
 For i = 0 To 4
 If i = sd.currentPanel Then
  imgs(i).Background = cd2
  Else
  imgs(i).Background = cd
 End If
 Next
End Sub


Sub ChangePanel(Left As Boolean)
	SlidingPanels.ChangePanel(sd, Left)
End Sub

Sub Animation1_AnimationEnd
	SlidingPanels.AnimationEnd(sd)
	If sd.targetPanel >= 0 Then
		tmrAnimation.Enabled = True
		Return
	End If
End Sub
Sub tmrAnimation_Tick
	tmrAnimation.Enabled = False
	ContinueJumping
End Sub

Sub JumpToPanel (Target As Int)
	sd.targetPanel = Target
	For i = 0 To 1
		sd.leftAnimations(i).Duration = SlidingDuration / 2
		sd.rightAnimations(i).Duration = SlidingDuration / 2
	Next
	ContinueJumping
End Sub
Sub ContinueJumping
	If sd.targetPanel < 0 Or sd.targetPanel = sd.currentPanel Then
		sd.targetPanel = -1
		Animation1_AnimationEnd
		For i = 0 To 1
			sd.leftAnimations(i).Duration = SlidingDuration
			sd.rightAnimations(i).Duration = SlidingDuration
		Next
		Return
	End If
	SlidingPanels.ChangePanel(sd, sd.targetPanel > sd.currentPanel)
End Sub
```

```
Sub Panels_Touch (Action As Int, X As Float, Y As Float)
	Select Action
		Case pContent.ACTION_DOWN
			startX = X
			startY = Y
		Case pContent.ACTION_UP
			If Abs(Y - startY) > 20%y Then Return
			If X - startX > 30%x  Then
				ChangePanel(False)
			Else If startX - X > 30%x Then
				ChangePanel(True)
			End If
	End Select
End Sub


Sub ActionBar_NavigationItemClick
  Activity.Finish
End Sub
```

About

```
#Region  Activity Attributes
	#FullScreen: False
	#IncludeTitle: True
#End Region
#Extends: android.support.v7.app.AppCompatActivity
Sub Process_Globals
	'These global variables will be declared once when the application starts.
	'These variables can be accessed from all modules.


End Sub

Sub Globals
	'These global variables will be redeclared each time the activity is created.
	'These variables can only be accessed from this module.

	Dim AC As AppCompat
	Dim ABHelper As ACActionBar
	Private pContent As Panel
Private ActionBar As ACToolBarLight

	Dim scvTest As ScrollView
	Dim pnlTest As Panel
End Sub
```

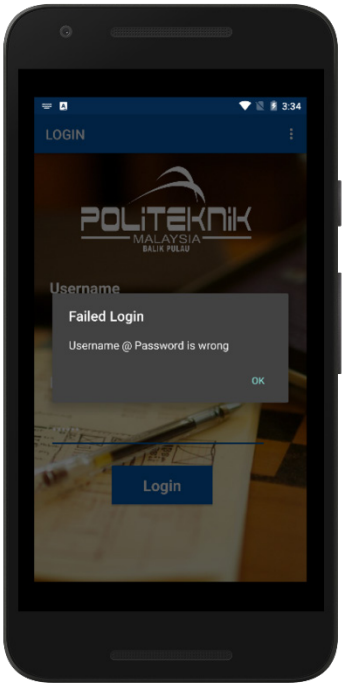
```
Sub Activity_Create(FirstTime As Boolean)

Activity.LoadLayout("main")

	scvTest.Panel.LoadLayout("about")
	scvTest.Panel.Height = pnlTest.Height




		ActionBar.Title = "ABOUT JSKK"
	ActionBar.SubTitle = ""
	ABHelper.Initialize
ABHelper.ShowUpIndicator = True
ActionBar.InitMenuListener


End Sub

Sub Activity_Resume

End Sub

Sub Activity_Pause (UserClosed As Boolean)

End Sub

Sub ActionBar_NavigationItemClick
  Activity.Finish
End Sub

Sub edtItem_FocusChanged (HasFocus As Boolean)
	Dim Send As EditText

	If HasFocus Then
		Send = Sender

		scvTest.ScrollPosition = Send.Top - 10dip
	End If
End Sub
```

PRINTSCREEN

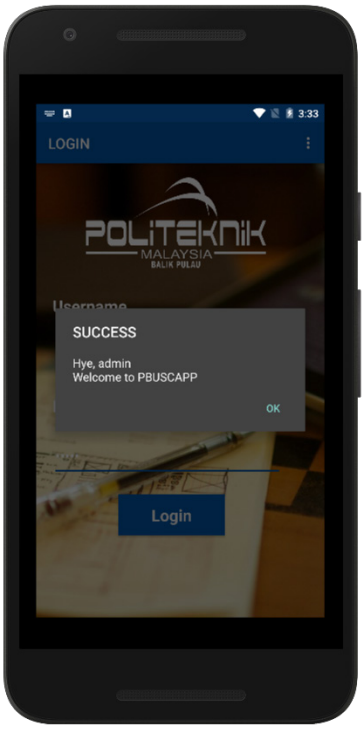
- **Loading Screen**

- **Login Screen**

- **If Username @ Password is Wrong**



- **If success**

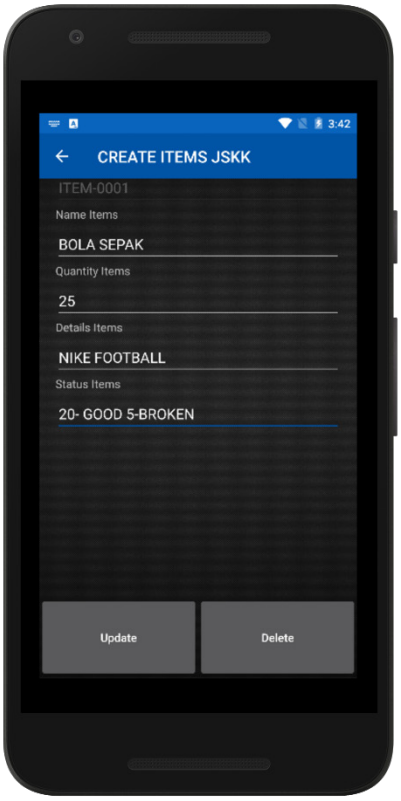

3:33
LOGIN
POLITEKNIK
MALAYSIA
BALIK PULAU
SUCCESS
Hye, admin
Welcome to PBUSCAPP
OK
Login

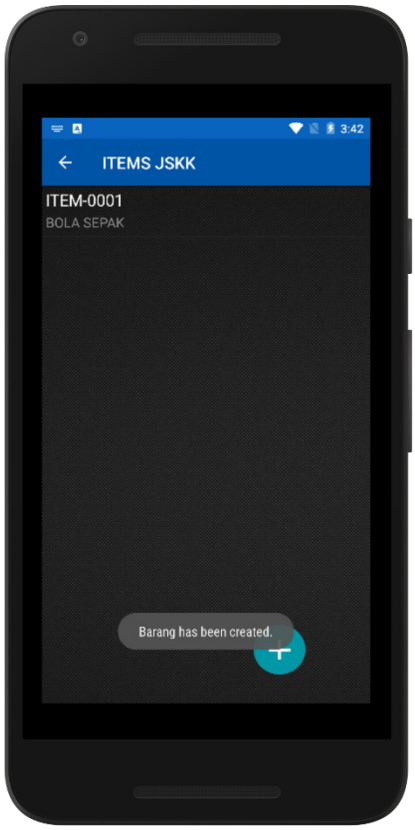
- **Home**


3:35
MENU
POLITEKNIK
MALAYSIA
BALIK PULAU
Item button
Gallery Button
About button
Log out button

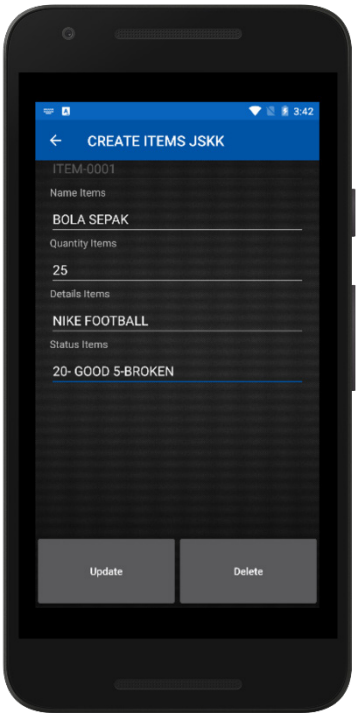
- **Button Item**

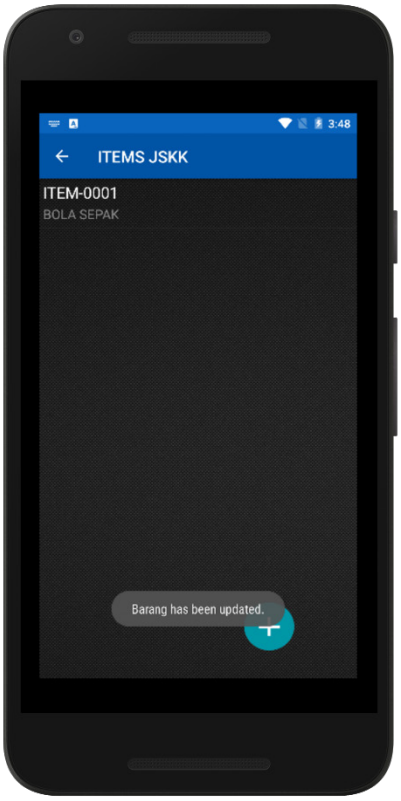

3:40
ITEMS JSKK


3:42
CREATE ITEMS JSKK
ITEM-0001
Name Items
BOLA SEPAK
Quantity Items
25
Details Items
NIKE FOOTBALL
Status Items
20- GOOD 5-BROKEN
Update
Delete

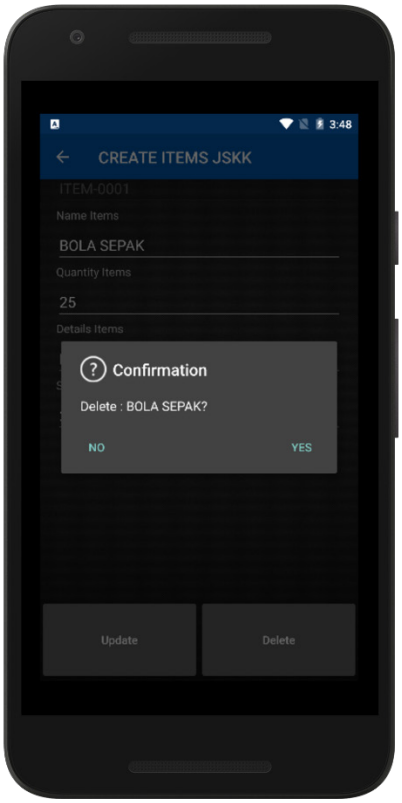


- **Update Item**


3:42
CREATE ITEMS JSKK
ITEM-0001
Name Items
BOLA SEPAK
Quantity Items
25
Details Items
NIKE FOOTBALL
Status Items
20- GOOD 5-BROKEN
Update
Delete


3:48
ITEMS JSKK
ITEM-0001
BOLA SEPAK
Barang has been updated.

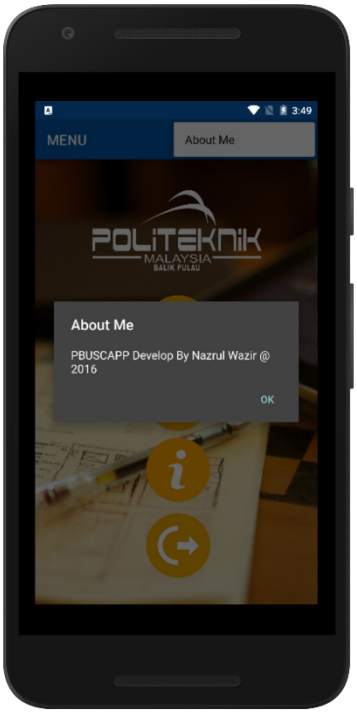
- **Delete Item**


3:48
CREATE ITEMS JSKK
ITEM-0001
Name Items
BOLA SEPAK
Quantity Items
25
Details Items
Confirmation
Delete : BOLA SEPAK?
NO
YES
Update
Delete

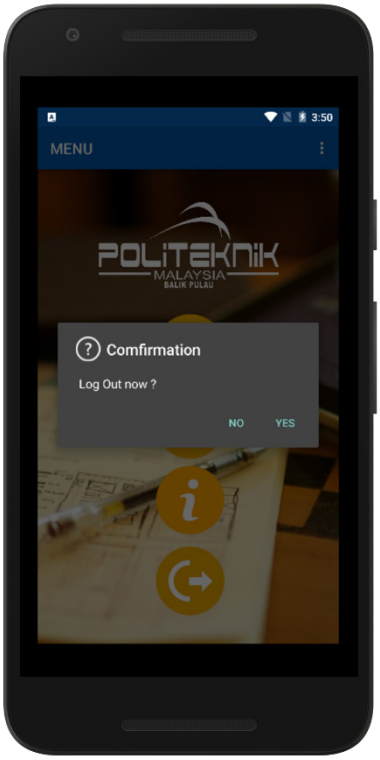
- **Gallery Button**

- **About Button**

- **About me (Menu Item)**



- **Log Out Button**



CONCLUSION

As a conclusion of the result, we should mention that most of the sports center staff prefer their sports storage stored data more effectively and safely. They do not have to waste time doing work hard. So just using Mobile Application is easy for them to do the job. So hope this app will benefit the public.

Last, I would like to thanks to our friends and lecturer as we have also got a lot of helps from them. And thanks to the PBU dormitory officers and students there, because they have responded well to us of our questions.

REFERENCE

1. PUAN NORHALIZA BINTI IDRIS

(Lecturer of Mobile Application)

(POLITEKNIK BALIK PULAU)

2. Official Portal Politeknik Balik Pulau

http://www.pbu.edu.my/

3. Online Web Tutorial about B4A

www.b4x.com

4. Solution about error

http://stackoverflow.com/