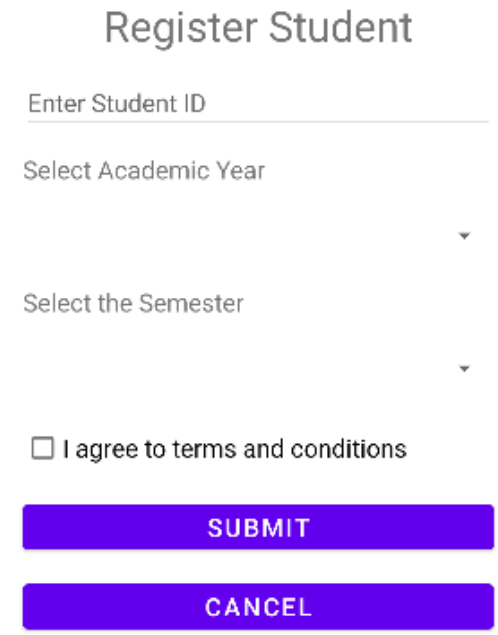
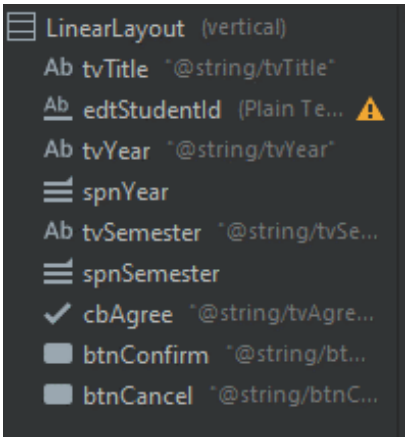
IT2010 – Mobile Application Development
BSc (Hons) in Information Technology
2nd Year
Faculty of Computing
SLIIT
2023 - Tutorial

-From Validation

1. Create the following UI





Important

- Layout – Linear Layout (Vertical)
- spnYear – Spinner
- Margins – 16dp (Top, Left, and Right)
- cbAgree – Check box
- Label Text size – 20sp
- Title Text size – 34sp

Strings

```
<string name="app_name">My Application</string>
<string name="tvTitle">Register Student</string>
<string name="edtStudentID">Enter Student ID</string>
<string name="tvYear">Select Academic Year</string>
<string name="tvSemester">Select the Semester</string>
<string name="btnSubmit">Submit</string>
<string name="btnCancel">Cancel</string>
<string name="cbAgreement">I agree to terms and conditions</string>

<string-array name="years">
    <item></item>
    <item>1</item>
    <item>2</item>
    <item>3</item>
    <item>4</item>
</string-array>

<string-array name="semesters">
    <item>Semester 1</item>
    <item>Semester 2</item>
</string-array>
```

2. For the spinners, set the entries attribute with the respective string array.
3. Create a new package named ‘models’

4. Create a new package named validations within the ‘models’ package
5. Create a new sealed class named ValidationResult

```
sealed class ValidationResult{
    data class Empty(val errorMessage:String):ValidationResult()
    data class Invalid(val errorMessage: String):ValidationResult()
    object Valid:ValidationResult()
}
```

6. Create a new class called FormData in models directory

```
class FormData(
    private var studentID:String,
    private var year:String,
    private var semester:String,
    private var agree:Boolean,
) {
    fun validateStudentId(): ValidationResult {
        return if(studentID.isEmpty()){
            ValidationResult.Empty("Student ID is empty")
        }else if(!studentID.startsWith("IT")){
            ValidationResult.Invalid("Should be starting with IT")

        }else if(studentID.length<10){
            ValidationResult.Invalid("Student ID should have 10 characters")
        }else if(studentID.length>10){
            ValidationResult.Invalid("Student ID should have 10 characters")
        }else{
            ValidationResult.Valid
        }

    }

    fun validateYear(): ValidationResult {
        return if(year.isEmpty()){
            ValidationResult.Empty("Year is empty")
        }else{
            ValidationResult.Valid
        }
    }

    fun validateSemester(): ValidationResult {
        return if(semester.isEmpty()){
            ValidationResult.Empty("Semester is empty")
        }else{
            ValidationResult.Valid
        }
    }

    fun validateAgreement():ValidationResult{
```

```
        return if(!agree){
            ValidationResult.Invalid("You must agree for the terms and
conditions")
        }else{
            ValidationResult.Valid
        }
    }

}
```

7. In the MainActivity implement the following code.

Global variables for the views

```
lateinit var edtStudentId:EditText
lateinit var spnYear: Spinner
lateinit var spnSemester:Spinner
lateinit var cbAgree:CheckBox
lateinit var tvYear:TextView
lateinit var tvSemester:TextView
private var count = 0;
```

- Observe the usage of the count variable

initialization of the views

```
edtStudentId  = findViewById(R.id.edtStudentId)
tvYear = findViewById(R.id.tvYear)
spnYear = findViewById(R.id.spnYear)
tvSemester = findViewById(R.id.tvSemester)
spnSemester = findViewById(R.id.spnSemester)
cbAgree = findViewById(R.id.cbAgree)
```

implement the displayAlert function

```
fun displayAlert(title:String, message:String){
    val builder = AlertDialog.Builder(this)
    builder.setTitle(title)
    builder.setMessage(message)
    builder.setPositiveButton("OK") { dialog, which ->
        // Do something when the "OK" button is clicked
    }
    val dialog = builder.create()
    dialog.show()
}
```

implement the submit function

```
fun submit(v: View){
    val myForm = FormData(
        edtStudentId.text.toString(),
        spnYear.selectedItem.toString(),
        spnSemester.selectedItem.toString(),
```

```
        cbAgree.isChecked
    )

    val studentIdValidation = myForm.validateStudentId()
    val spnYearValidation = myForm.validateYear()
    val spnSemesterValidation = myForm.validateSemester()
    val cbAgreeValidation = myForm.validateAgreement()

    when(studentIdValidation){
        is ValidationResult.Valid ->{
            count ++
        }
        is ValidationResult.Invalid ->{
            edtStudentId.error = studentIdValidation.errorMessage
        }
        is ValidationResult.Empty ->{
            edtStudentId.error = studentIdValidation.errorMessage
        }
    }

    when(spnYearValidation){
        is ValidationResult.Valid ->{
            count ++
        }
        is ValidationResult.Invalid ->{
           val tv:TextView = spnYear.selectedView as TextView
            tv.error =""
            tv.text = spnYearValidation.errorMessage
        }
        is ValidationResult.Empty ->{
            val tv:TextView = spnYear.selectedView as TextView
            tv.error =""
            tv.text = spnYearValidation.errorMessage
        }
    }

    when(spnSemesterValidation){
        is ValidationResult.Valid ->{
            count ++
        }
        is ValidationResult.Invalid ->{
            val tv:TextView = spnSemester.selectedView as TextView
            tv.error =""
            tv.text =  spnSemesterValidation.errorMessage
        }
        is ValidationResult.Empty ->{
            val tv:TextView = spnSemester.selectedView as TextView
            tv.error =""
            tv.text =  spnSemesterValidation.errorMessage

        }
    }

    when(cbAgreeValidation){
        is ValidationResult.Valid ->{
            count ++
        }
```

```
        is ValidationResult.Invalid ->{
            displayAlert("Error", cbAgreeValidation.errorMessage)
        }
        is ValidationResult.Empty ->{

        }
    }

    if(count==4){
        displayAlert("Success","You have successfully registered")
    }
}
```

8. Run the app and test all the scenarios