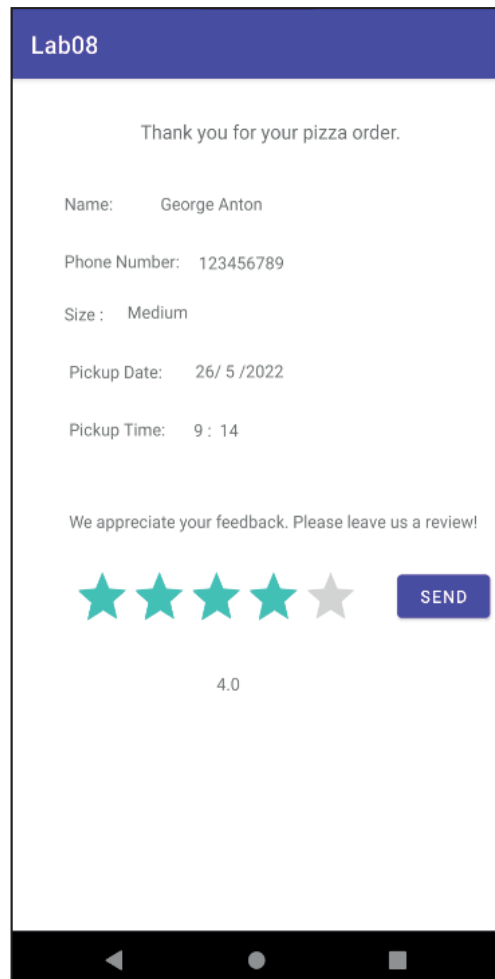
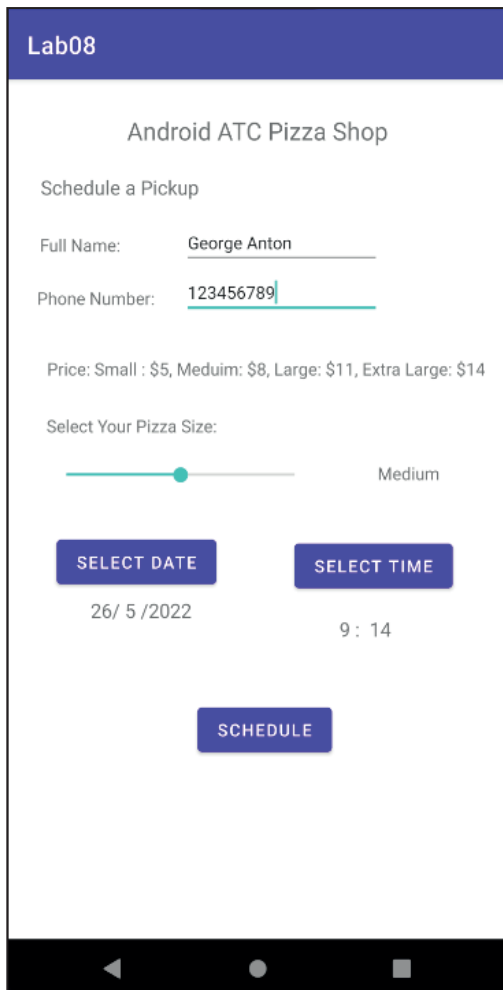


Lab 8

Creating a Pizza Schedule Pickup App

In this lab you will create a pizza shop schedule pickup app. The app user may use this app to schedule pickup for his/her pizza order. In this app you will achieve the following tasks:

- Configuring the Pizza size using SeekBar widget.
- Configuring the order Pickup date using the Date Picker Class
- Configuring the order Pickup time using the Time Picker Class
- Pass the app order details to another activity using the Intent class.
- Using the RatingBar widget to leave the app user review.



To create this pizza schedule pickup app, follow the following steps:

- 1- Open Android Studio, and then click **File** → **New** → **New Project**
- 2- Select **Empty Activity**, and click **Next**
- 3- Type: **Lab08** for the application name, then click **Finish**.
- 4- Before you start with typing the Kotlin code in your app, check the **build.gradle (Module: Lab08.app)** file content, and be sure it has the Kotlin plugin. If not, add the following code:
id 'kotlin-android-extensions'

And click the **Sync Now**. The configuration should be as follows:

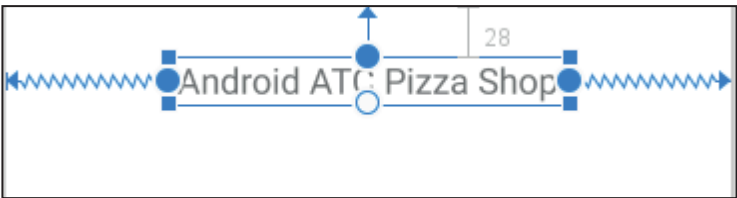
```

1  plugins {
2      id 'com.android.application'
3      id 'kotlin-android'
4      id 'kotlin-android-extensions'
5  }
```

- 5- Open the **activity_main.xml** file in the **Design** mode, and then **delete** the “Hello World!” **text**.
- 6- Add a **TextView** widget to your activity, set its constraints, and set its attributes values as follows:

text: Android ATC Pizza Shop	textSize: 20sp
-------------------------------------	-----------------------

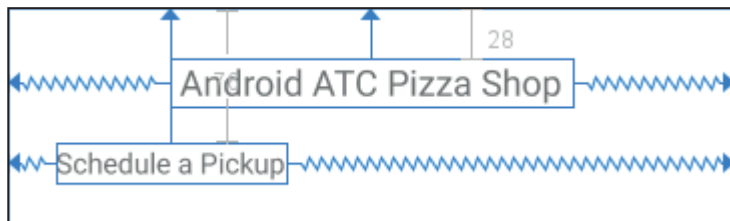
Your activity should have the following design:



- 7- Add a **TextView** widget to your activity, set its constraints, and set its attributes values as follows:

text: Schedule a Pickup	textSize: 16sp
--------------------------------	-----------------------

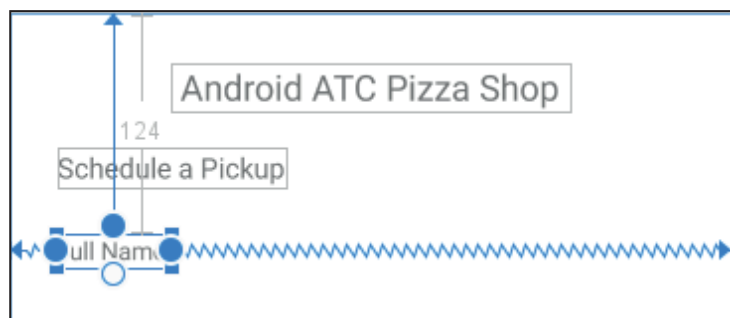
Your activity should have the following design:



8- Add a **TextView** widget to your activity, set its constraints, and set its attributes values as follows:

text: Full Name:	textSize: 14sp
-------------------------	-----------------------

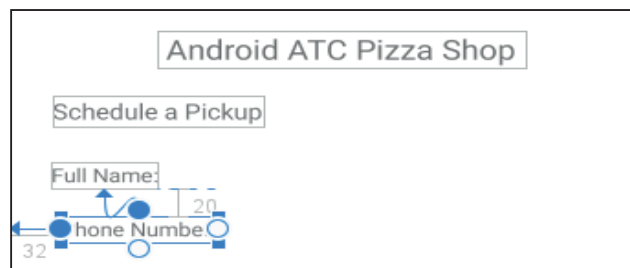
Your activity should have the following design:



9- Add a **TextView** widget to your activity, set its constraints, and set its attributes values as follows:

text: Phone Number:	textSize: 14sp
----------------------------	-----------------------

Your activity should have the following design:



10- Add a **Plain Text** widget to your activity, place it beside the **Full Name** TextView widget, set its constraints, and set its attributes values as follows:

id: myFullName	textSize: 14sp
-----------------------	-----------------------

Delete the **text** attribute value.

Your activity should have the following design:

The design view shows a form titled "Android ATC Pizza Shop". Below the title is a button labeled "Schedule a Pickup". Underneath the button are two text input fields. The first field is preceded by the label "Full Name:" and the second field is preceded by the label "Phone Number:". The "Phone Number:" label is highlighted with a blue border.

11- Add a **Plain Text** widget to your activity, place it beside the **Phone Number** *TextView* widget, set its constraints, and set its attributes values as follows:

id: myPhoneNumber	textSize: 14sp
--------------------------	-----------------------

Delete the **text** attribute value.

Your activity should have the following design:

The design view shows the same form as before, but now the "Phone Number:" label is followed by a text input field. The "Schedule a Pickup" button is still present above the input fields.

12- Add **TextView** widget, set its constraints, and set its attributes values as follows:

text: Select Your Pizza Size:	textSize: 14sp
--------------------------------------	-----------------------

Your activity should have the following design:

The design view shows the form with the "Select Your Pizza Size:" label added at the bottom. The layout is annotated with blue dimension lines and numbers indicating the spacing and alignment of the widgets. The "Schedule a Pickup" button is at the top, followed by the "Full Name:" and "Phone Number:" input fields. The "Select Your Pizza Size:" label is at the bottom, with a text input field to its right.

13- Add a **SeekBar** widget from the **Palette** panel (Widgets) to your activity.

Set the **SeekBar** constraints and configure its attributes values as follows:

id = mySeekBar	layout_width= 220dp	layout_height= 20dp	max= 4
-----------------------	----------------------------	----------------------------	---------------

Your activity should have the following design:

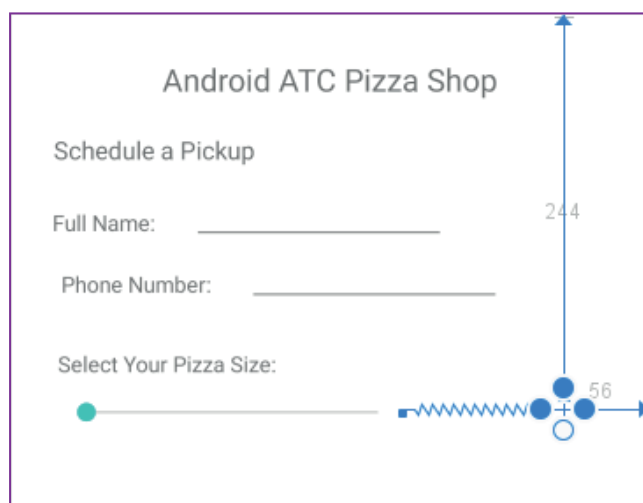


14- Add a **TextView** widget to your activity to the right side of your **SeekBar**, set its constraints, and set its attributes values as follows:

id: myPizzaSize	textSize: 14sp
------------------------	-----------------------

Delete the **text** attribute value.

Your activity should have the following design:



This **TextView** widget will be used to display the SeekBar value.

15- To configure your **SeekBar** to display the four pizza sizes: Small, Medium, Large, and Extra-Large, open the **MainActivity** file.

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        lateinit var slider: SeekBar
        lateinit var value: TextView

        slider=mySeekBar
        value=myPizzaSize

    }
}
```

16- Double click the **SeekBar**, click the red pop-up lamp, and select **Import**

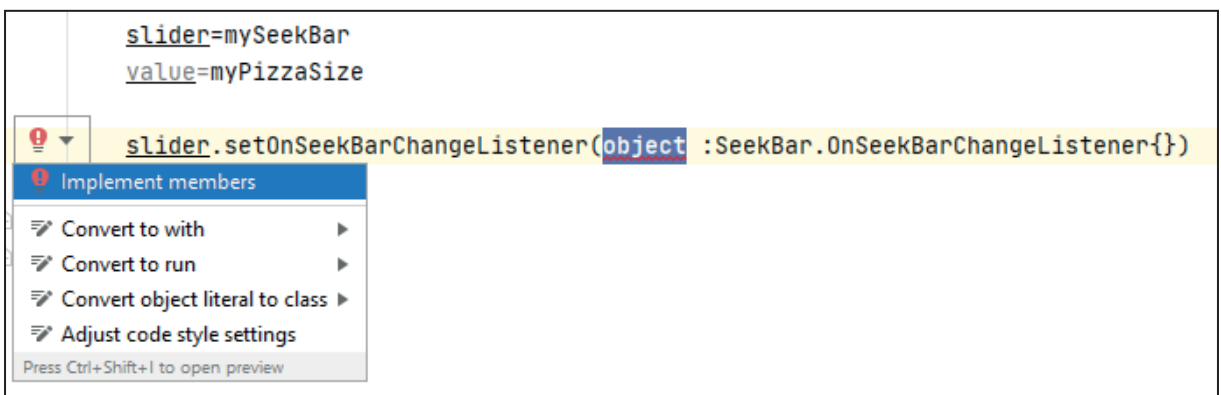
17- Double click the **TextView**, click the red pop-up lamp, and select **Import**

18- Double click the **mySeekBar**, click the red pop-up lamp, and select **Import**

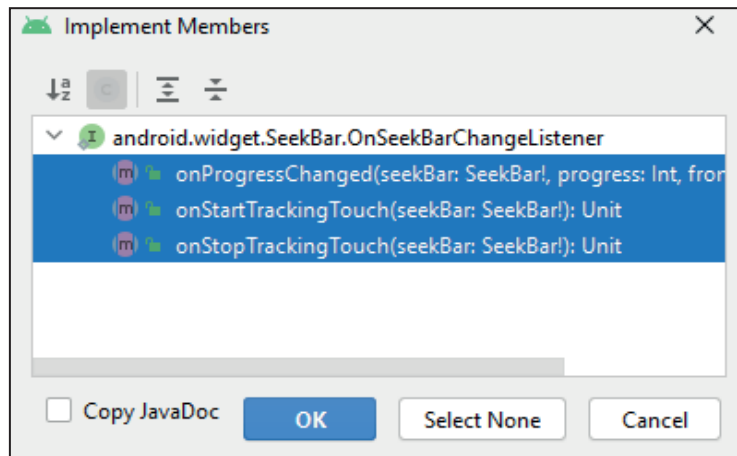
19- Add the following code to **MainActivity** file:

```
slider.setOnSeekBarChangeListener(object:SeekBar.OnSeekBarChangeListener{ } )
```

As illustrated in the following figure, double click **object**, and click the red pop-up lamp, and select **Implement members**



20- Press **Shift**, and select all the three methods as illustrated in the figure below, and click **OK**.



These three SeekBar methods will be added to your code as illustrated in the following figure:

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        lateinit var slider: SeekBar
        lateinit var value: TextView

        slider=mySeekBar
        value=myPizzaSize

        slider.setOnSeekBarChangeListener(object :SeekBar.
        OnSeekBarChangeListener{
            override fun onProgressChanged(seekBar: SeekBar?,
            progress: Int, fromUser: Boolean) {
                TODO("Not yet implemented")
            }

            override fun onStartTrackingTouch(seekBar: SeekBar?) {
                TODO("Not yet implemented")
            }

            override fun onStopTrackingTouch(seekBar: SeekBar?) {
                TODO("Not yet implemented")
            }
        })
    }
}
```

21- Remove the three **TODO** comments of these three **SeekBar** methods.

22- Now, you will configure the **onProgressChanged** method to display the progress level change in the **TextView** widget (**id**: myPizzaSize or value variable). The **max** attribute value of this **SeekBar** widget is: **4**. This means you can display 5 values with the start points.

Because you need to use a variable to display these 5 values, you should use the **ArrayList** class (For more information about the **ArrayList** class, review lesson 3 of this course.). Add this variable: **pizzaSize** to your code as follows:

```
val pizzaSize= arrayListOf<String>("Please Select","Small","Medium",
    "Large","Extra-Large")
```

Now, use this variable **pizzaSize** with **onProgressChanged** method to display the values of this array as illustrated in the following code:

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        lateinit var slider: SeekBar
        lateinit var value: TextView

        slider=mySeekBar
        value=myPizzaSize
        val pizzaSize= arrayListOf<String>("Please Select","Small",
    ,"Medium","Large","Extra-Large")

        slider.setOnSeekBarChangeListener(object:SeekBar.
        OnSeekBarChangeListener{
            override fun onProgressChanged(seekBar: SeekBar?, progress:
            Int, fromUser: Boolean) {

                value.text=pizzaSize[progress]

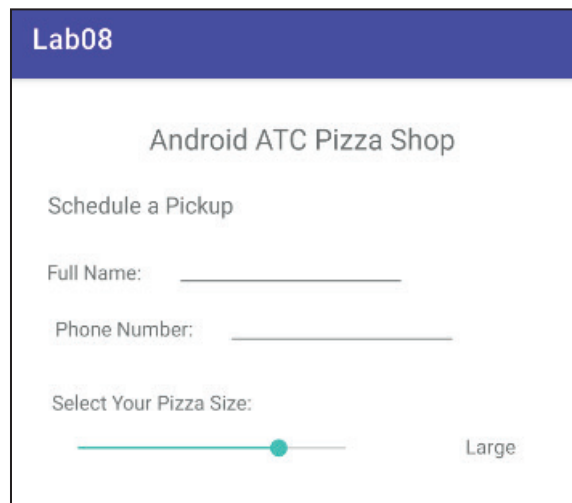
            }
            override fun onStartTrackingTouch(seekBar: SeekBar?) {

            }

            override fun onStopTrackingTouch(seekBar: SeekBar?) {

            }
        })
    }
}
```


23- **Stop**, then **Run** your app. Test your **SeekBar**. The following figure displays how the values change.



24- Now, you should configure the order pickup date and time for your app user. First, to configure the order pickup date, open the **activity_main.xml** in the **Design** mode, drag and drop **Button** widget to your activity, and set its constraints as illustrated in the following figure:



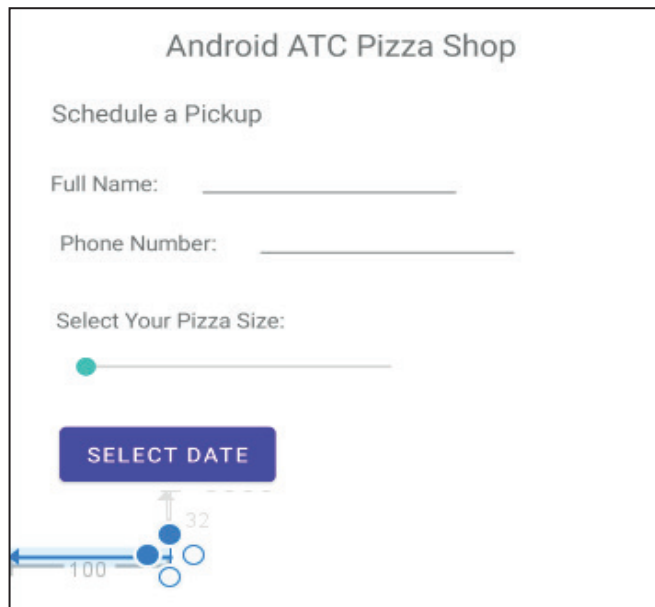
25- Configure this button attributes values as follows:

id: dateBtn	text: Select Date
--------------------	--------------------------

26- You need to add a **TextView** widget under the date button to display the selected date. Add this **TextView**, set its constraints, and set its attributes values as follows:

id: dateText	textSize: 16sp	layout_width: wrap_content	layout_height: wrap_content
---------------------	-----------------------	-----------------------------------	------------------------------------

Delete the **text** attribute value of this **TextView** or set it with a blank value. The activity will look like the figure below:

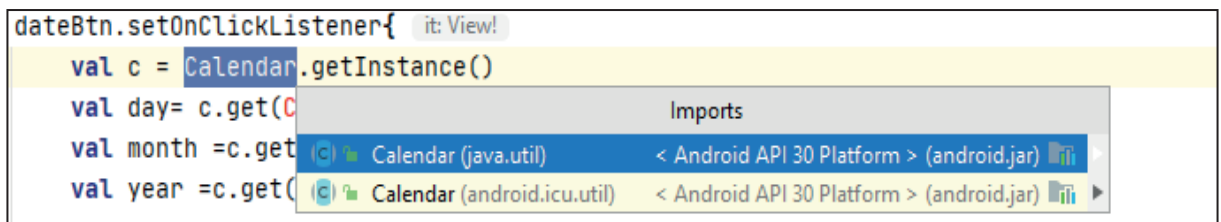


27- Now, you should configure this date button with the `DatePickerDialog` class. Open the **MainActivity** file, and add the following code:

```
dateBtn.setOnClickListener{
    val c = Calendar.getInstance()
    val day= c.get(Calendar.DAY_OF_MONTH)
    val month =c.get(Calendar.MONTH)
    val year =c.get(Calendar.YEAR)

    val myDatePicker = DatePickerDialog(this,android.R.style.
    ThemeOverlay,DatePickerDialog.OnDateSetListener {DatePicker,
    Year,Month,Day ->
        dateText.text="$Day/ ${Month+1} /$Year"},year,month,day)
    myDatePicker.show()
}
```

Double click the **Calendar** class, click the red pop-up lamp, and select **Import**. Then, as illustrated in the figure below select : **Calenda (java.util)**



Double click the **DatePickerDialog** class, click the red pop-up lamp, and select **Import**. The full code as follows:

```

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        lateinit var slider: SeekBar
        lateinit var value: TextView

        slider=mySeekBar
        value=myPizzaSize
        val pizzaSize= arrayListOf<String>("Please Select","Small",
"Medium","Large","Extra-Large")

        slider.setOnSeekBarChangeListener(object :SeekBar.
OnSeekBarChangeListener{
            override fun onProgressChanged(seekBar: SeekBar?,
progress: Int, fromUser: Boolean) {

                value.text=pizzaSize[progress]

            }
        })
        override fun onStartTrackingTouch(seekBar: SeekBar?) {

        }
        override fun onStopTrackingTouch(seekBar: SeekBar?) {

        }
    })

    dateBtn.setOnClickListener{
        val c = Calendar.getInstance()
        val day= c.get(Calendar.DAY_OF_MONTH)
        val month =c.get(Calendar.MONTH)
        val year =c.get(Calendar.YEAR)

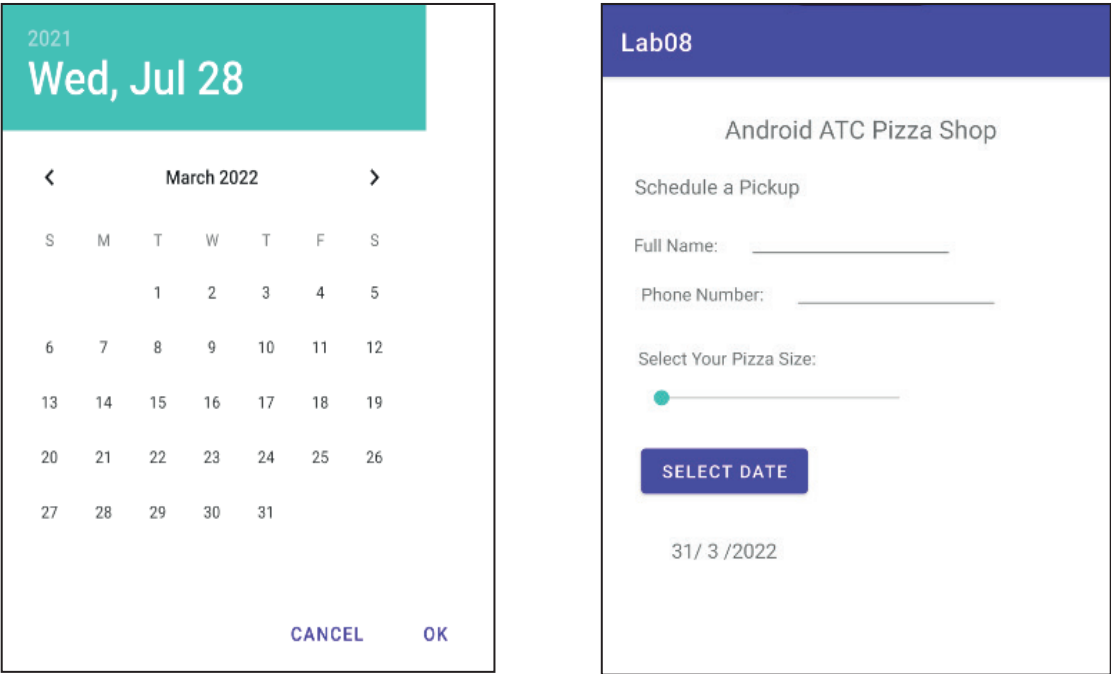
        val myDatePicker =
DatePickerDialog(this,android.R.style.
ThemeOverlay,DatePickerDialog.OnDateSetListener {DatePicker,
Year,Month,Day ->
            dateText.text="$Day/ ${Month+1} /$Year"},year,month,day)

            myDatePicker.show()

        }
    }
}

```

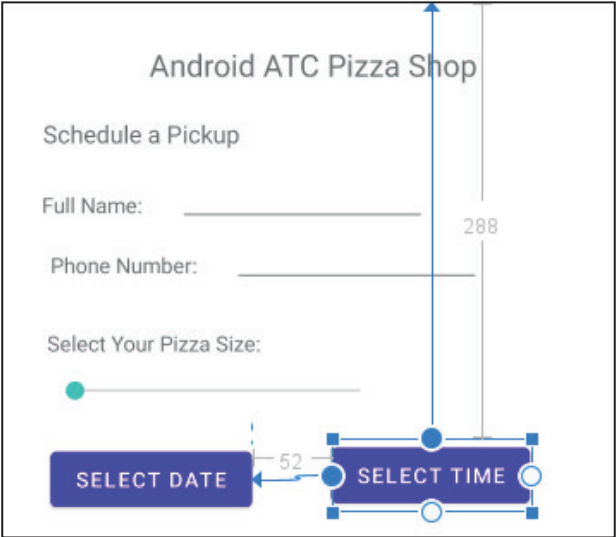
28- **Stop** your app, then **Run** it again. Test your **Select Date** button. The following figures display its run result:



29- Now, you should configure the pickup time in your app. Add a **Button** widget to your activity, set its constraints, and set its attributes values as follows:

id: timeBtn	text : Select Time
--------------------	---------------------------

Your activity interface design should be as follows:

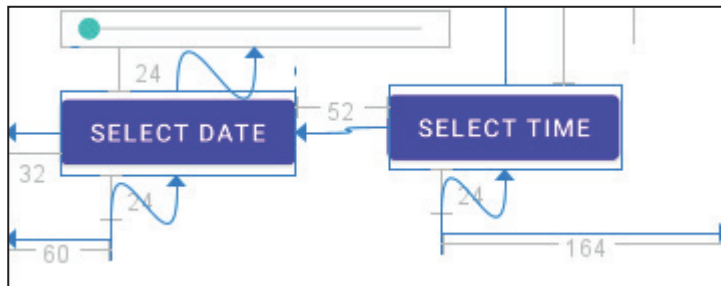


30- Add a **TextView** widget to your **activity_main.xml** file directly under the “SELECT TIME” button, Set its constraints, and then configure its attributes values as follows:

id: timeText	textSize: 16sp	Layout_width: wrap_content	Layout_height: wrap_content
---------------------	-----------------------	-----------------------------------	------------------------------------

Delete the **text** attribute value of this “TextView” or set it with a blank value.

The part of your activity interface should look like the figure below:



31- Open the **MainActivity** file and add the code below to create a time picker dialog directly under the previous date picker dialog code:

```
timeBtn.setOnClickListener{
    val c = Calendar.getInstance()
    val hour =c.get(Calendar.HOUR_OF_DAY)
    val minutes =c.get(Calendar.MINUTE)

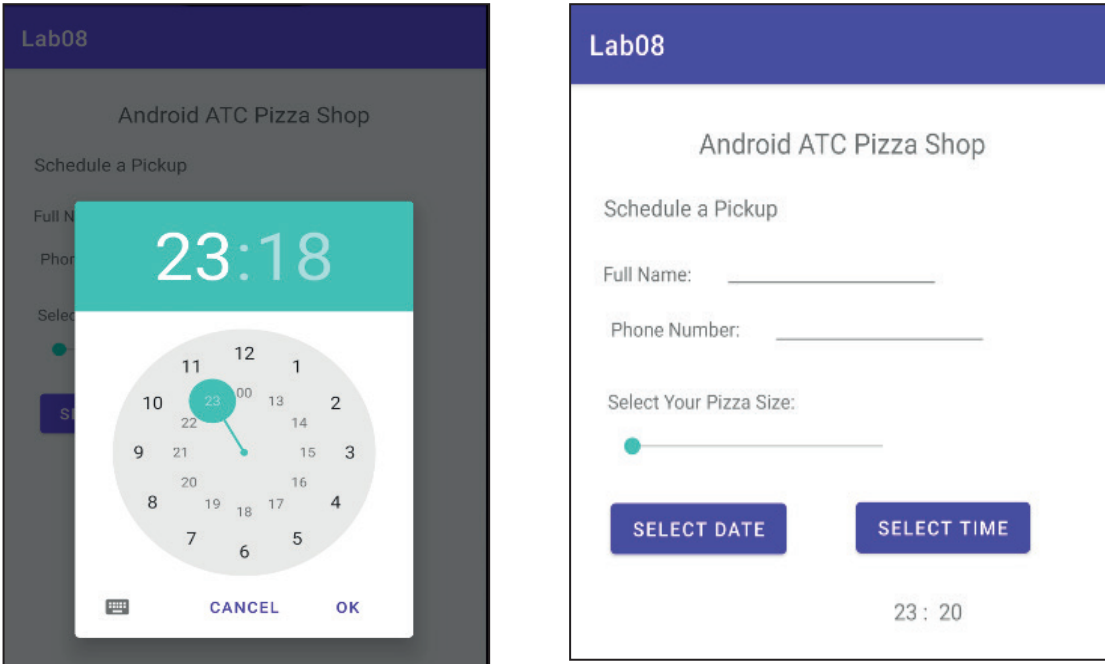
    val myTimePicker= TimePickerDialog(this,TimePickerDialog.
    OnTimeSetListener { TimePicker, hourOfDay, minute -> timeText.
    text="$hourOfDay : $minute"},hour,minutes,true)

    myTimePicker.show()
}
```

Double click the **TimePickerDialog**, click the red pop-up lamp, and select **Import**.

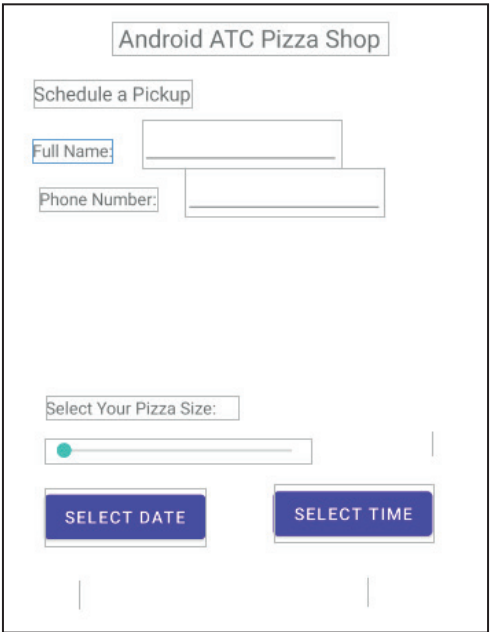
Note: the parameter “**true**” in the code above means that the time format is 24 hours.

32- **Stop**, and then **Run** your app. Tap the **SELECT TIME** button, select the time, then tap **OK**. . The run result should be as follows:



You may set the location of the `TextView` widget which displays the time many times to get it at a suitable location on your activity interface. The best way to get accurate location is by configuring this widget attributes values in the XML file such as `android:layout_marginTop`

33- It is good idea to display the price for each pizza size for your app users to know which size they should place an order for. You should add the price list as a **TextView** above the **SeekBar** calss. To do this, open the **activity_main.xml** file in **Design** mode, select the **SeekBar** and everthing is below it, then move it down a little as illustrated in the following figure:



34- Add to your activity a **TextView** widget, set its constraints, and set its **text** attribute value with :

Price: Small : \$5, Meduim: \$8, Large: \$11, Extra Large: \$14

35- Add a **Button** widget (Schedule buton) to your activity.This button should be configured to send all the app user data to the pizza shop cloud database, then a notification message will be sent to this pizza shop team to prepare the app user’s pizza and will guarantee it will be ready on the date and time which the app user selected.

Here in this lab, you will just send all this information using the Intent class to another activity.

Add a Button widget to your **activity_main.xml** activity, configure its constraints, and its attributes values as follows:

id : scheduleBtn	text : Schedule
-------------------------	------------------------

Your activity interface should have the following figure:

Android ATC Pizza Shop

Schedule a Pickup

Full Name:

Phone Number:

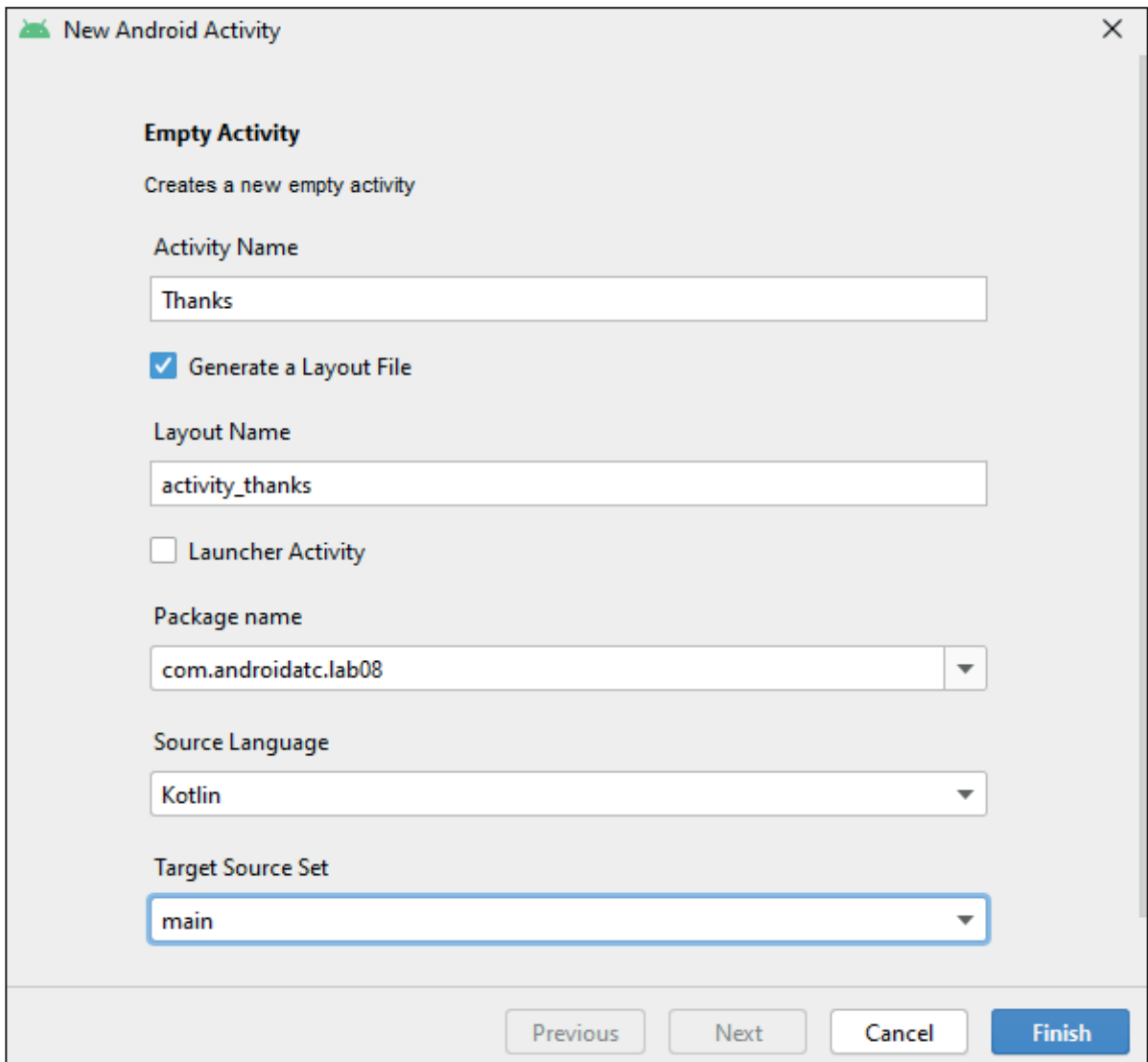
Price: Small : \$5, Meduim: \$8, Large: \$11, Extra Large: \$14

Select Your Pizza Size:

☒ ☐ ☐ ☐

36- To create a new activity for the app, right click **app** → **New** → **Activity** → **Empty Activity**

Type: **Thanks** for the **Activity Name** filed as illustrated in the figure below, keep the other default configurations and click **Finish**



New Android Activity

Creates a new empty activity

Activity Name
Thanks

☒ Generate a Layout File

Layout Name
activity_thanks

☐ Launcher Activity

Package name
com.androidatc.lab08

Source Language
Kotlin

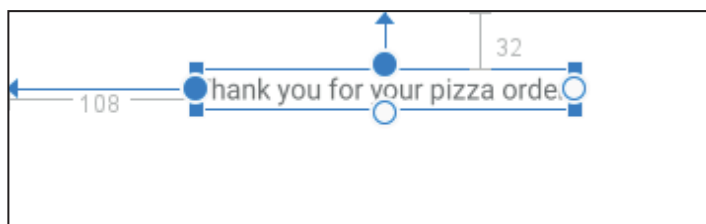
Target Source Set
main

Previous Next Cancel Finish

37- Open the **activity_thanks.xml** file in the **Design** mode. Add a **TextView** widget, set its constraints and set its attribute value **text** with:

text: Thank you for your pizza order.	textSize: 16sp
--	-----------------------

The **activity_thanks.xml** should have the following figure:



38- Open the **activity_thanks.xml** in the **Design** mode, add five **TextView** widgets, set their constraints and **text** attributes values as illustrated in the figure below:

Thank you for your pizza order.

Name:

Phone Number:

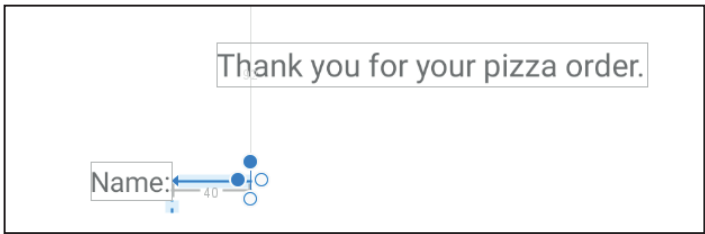
Size :

Pickup Date:

Pickup Time:

39- Now, in this **activity_thanks.xml**, you should add a **TextView** widget beside the “**Name:**” to display the name which will be typed by the app user in the main activity. This **TextView** widget which you will be added in the **activity_thanks.xml** must have an empty (blank) **text** attribute value as illustrated in the figure below:

Set the **id** attribute value of this **TextView** : **nameld**



40- Repeat the same previous step to add a **TextView** with empty **text** attribute value for each file of the app user data. Set their constraints and configure their **id** attributes values as the following:

id: phoneld	id: sizeld	id: dateld	id: timeld
-------------	------------	------------	------------

You should have the following activity interface for the **activity_thanks.xml**

Thank you for your pizza order.

Name: |

Phone Number: |

Size : |

Pickup Date: |

Pickup Time: |

41- Open the **MainActivity** file, and configure the **Intent** class with **putExtra** method to pass the app user data from the **MainActivity** to the **Thanks** activity.

Type the following code in the **MainActivity** file:

```
scheduleBtn.setOnClickListener{
    var intent = Intent(this,Thanks::class.java)
    intent.putExtra("name", myFullName.text.toString())
    intent.putExtra("phoneNumber", myPhoneNumber.text.toString())
    intent.putExtra("pizzSize", myPizzaSize.text.toString())
    intent.putExtra("pickupDate", dateText.text.toString())
    intent.putExtra("pickupTime", timeText.text.toString())

    startActivity(intent)
}
```

In the code above, double click the **Intent** class, click the red pop-up lamp, and select **Import**.

The full code of the **MainActivity** file is as follows:

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        lateinit var slider: SeekBar
        lateinit var value: TextView

        slider=mySeekBar
        value=myPizzaSize
        val pizzaSize= arrayListOf<String>("Please Select","Small",
        "Medium","Large","Extra-Large")

        slider.setOnSeekBarChangeListener(object :SeekBar.
        OnSeekBarChangeListener{
            override fun onProgressChanged(seekBar: SeekBar?,
            progress: Int, fromUser: Boolean) {

                value.text=pizzaSize[progress]

            }
            override fun onStartTrackingTouch(seekBar: SeekBar?) {

            }
            override fun onStopTrackingTouch(seekBar: SeekBar?) {

            }
        })
    }
}
```

```

        dateBtn.setOnClickListener{
            val c = Calendar.getInstance()
            val day= c.get(Calendar.DAY_OF_MONTH)
            val month =c.get(Calendar.MONTH)
            val year =c.get(Calendar.YEAR)

            val myDatePicker =
DatePickerDialog(this,android.R.style.
ThemeOverlay,DatePickerDialog.OnDateSetListener {DatePicker,
Year,Month,Day ->
            dateText.text="$Day/ ${Month+1}/${Year}" ,year,month,day)

            myDatePicker.show()

        }

        timeBtn.setOnClickListener{
            val c = Calendar.getInstance()
            val hour =c.get(Calendar.HOUR_OF_DAY)
            val minutes =c.get(Calendar.MINUTE)

            val myTimePicker=
TimePickerDialog(this,TimePickerDialog.OnTimeSetListener {
TimePicker, hourOfDay, minute -> timeText.text="$hourOfDay :
$minute"},hour,minutes,true)

            myTimePicker.show()

        }

        scheduleBtn.setOnClickListener{
            var intent  = Intent(this,Thanks::class.java)
            intent.putExtra("name", myFullName.text.toString())
            intent.putExtra("phoneNumber", myPhoneNumber.text.toString())
            intent.putExtra("pizzSize", myPizzaSize.text.toString())
            intent.putExtra("pickupDate", dateText.text.toString())
            intent.putExtra("pickupTime", timeText.text.toString())

            startActivity(intent)
        }
    }
}

```

Note: In the previous code you used the **id** attribute value of the widgets which have used to display the data. For example, you used the **id** of the **TextView** widget which you have used to display the date pickup (**id**: dateText) not the id of the Select Date. The same thing for all other data.

Also, in the code above, the first part of the **putExtra** methods (name, phoneNumber, pizzSize, pickupDate, and pickupTime) must be the same with the **getStringExtra** method which you will use in the Thanks activity to retrieve the data.

42- Open the **Thanks** file, and configure the **Intent** class with **getStringExtra** method to retrieve the app user data form the **MainActivity** file.

Type the following code in the **Thanks** file:

```
class Thanks : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_thanks)
        nameId.text=intent.getStringExtra("name")
        phoneId.text=intent.getStringExtra("phoneNumber")
        sizeId.text=intent.getStringExtra("pizzSize")
        dateId.text=intent.getStringExtra("pickupDate")
        timeId.text=intent.getStringExtra("pickupTime")
    }
}
```

In the code above, double click **nameId**, click the pop-up red lamp, and select **Import**.

43- Stop, then Run your app. Fill out the pizza pickup form, then tap the Schedule button. You should pass all your data into the Thanks activity as illustrated in the following figures:

Lab08

Android ATC Pizza Shop

Schedule a Pickup

Full Name:

Phone Number:

Price: Small : \$5, Meduim: \$8, Large: \$11, Extra Large: \$14

Select Your Pizza Size:

Medium

7/ 12 /2021

9 : 0

Lab08

Thank you for your pizza order.

Name:

Phone Number:

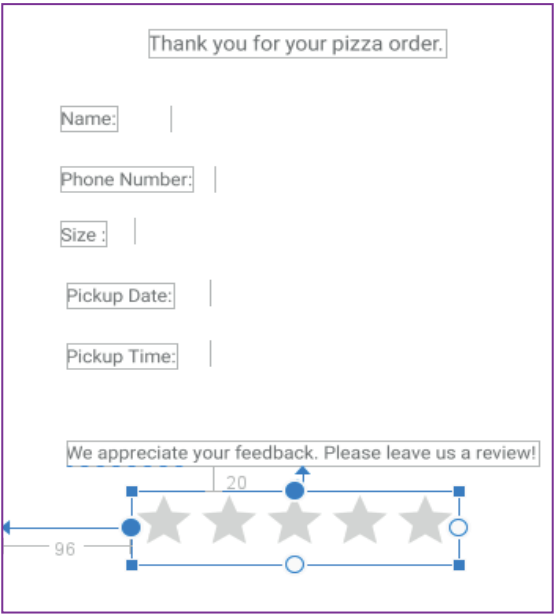
Size :

Pickup Date:

Pickup Time:

44- The company which uses your app may ask its app users (customers) to send their feedback or review. Let us use the **RatingBar** widget in the **Thanks** activity to add the review or feedback service. To do that, open the **activity_thanks.xml** file in the **Design** mode, add a **TextView** widget, set its constraints, and see its attributes values as follows: **text**: We appreciate your feedback. Please leave us a review!

45- Add a **RatingBar** widget, set its constraints and configure its attributes values as follows:



Set its attributes values as follows: **id**: myRatingBar and **numStars**: 5

46- Using drag and drop technique, add a **Button** widget to your Thanks activity beside the RatingBar widget, and set its constraints. Configure this button widget attributes values as follows:

id: sendBtn	text: Send
--------------------	-------------------

47- Add a **TextView** widget under the **RatingBar** widget and set its constraints. You will use this **TextView** widget later to display the rating value which will be selected by the app user.

The Thanks activity interface should have the following design:

Set the attributes values of the **TextView** widget as follows:

id: rateText	layout_width: wrap_content	layout_height: wrap_content
---------------------	-----------------------------------	------------------------------------

Also, delete the **text** attribute value.

48- Open the **Thanks** file and configure your **Send** button (id: sendBtn) to display the rating value using the **getRating()** method in the **TextView** widget (id :rateText). The code is as follows:

```
class Thanks : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_thanks)
        nameId.text=intent.getStringExtra("name")
        phoneId.text=intent.getStringExtra("phoneNumber")
        sizeId.text=intent.getStringExtra("pizzSize")
        dateId.text=intent.getStringExtra("pickupDate")
        timeId.text=intent.getStringExtra("pickupTime")

        sendBtn.setOnClickListener {
            rateText.text = myRatingBar.rating.toString()
        }
    }
}
```

49- **Stop**, then **Run** your app. Tap the **Schedule** button, select your rating level, and tap the **Send** button. If you select three stars, the run result should be as illustrated in the figure below:

