Ex No: 12

MOBILE APPLICATION TO SEND EMAIL

Aim:

To design and develop a front-end Android application that allows users to compose and send emails by collecting user input for recipient address, subject, and message, and providing a user-friendly interface with validation and interaction handling.

Algorithm:

l. Design the User Interface

- Create an input field for entering the recipient's email address.
- Add an input field for the email subject.
- Include a multi-line text area for the message body.
- Add a "Send" button to initiate the email sending process.
- Optionally, include success or error message display areas (e.g., Toast or Snackbar).

2. Input Collection Phase

-Capture user inputs from the recipient, subject, and message fields.

- Validate the recipient email address format.
- Enable the "Send" button only when all fields are filled correctly.

3. Action Trigger Phase

- Detect the "Send" button click event.
- Retrieve all input values.
- Pass input data to the email logic component (not implemented in the front end).

4. User Feedback Phase

- Display confirmation message upon successful triggering of the email intent.
- Show error messages if fields are empty or email format is invalid.
- Optionally reset the input fields after sending. Security and UX Considerations - Disable the send button during processing to prevent multiple taps.
- Avoid storing sensitive email data locally.
- Ensure email inputs are trimmed and sanitized.
- Prevent sending to malformed or malicious addresses.
- Maintain responsive and clean UI with accessible design.

CODE:

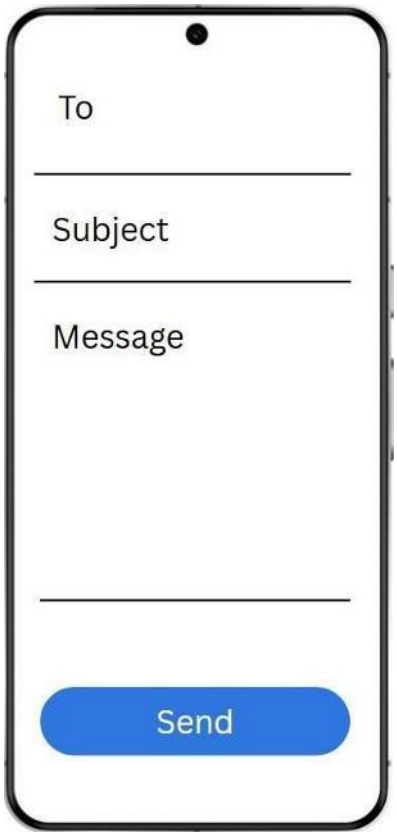
KOTLIN:

```
// Main Activity package com .
example . emailsender
import android. content. Intent android. net
import . Uri android. os . Bundle androidx .
import appcompat . app . AppCornpatAct _ivity
import android. widget . Button com . example
import . email sender . R
ilnport
class MainActivity : AppCompatActivity ( ) {override
    fun onCreate (savedInstanceState: Bundle?) {super
    . onCreate (saved InstanceState) setContentView
    (R. layout . activity main)

        val    sendEmai = findViewById<Button> 1Btn(R.     _id     .
        sendEmai1Btn)   stener {
sendEmai1Btn. setOnC1ickListener val recipient — arrayOf
           ("recipient@example . com") //
Replace with actual email val subj ect — Hello from my app" val
body = "This is a test email sent from my Android app .

           val intent Intent (Intent. ACTION SENDTO) . apply { data
— Uri . parse ("mailto: 'l ) // Only email apps should handle this
putExtra (Intent. EXTRA EMAIL, recipient) putExtra (Intent. EXTRA
SUBJECT, subj ect) putExtra (Intent. EXTRA TEXT, body)


           if ( intent . resolveActivity (packageManager)      null) {
               startActivity ( intent)
```

XML:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns : android— "http : / / schemas . android
.  com/apk/res/android"  android:  layout  parent"  android:
layout parent "
    android: " android: gravity="
    center" android: padding=" 16dp
    " >
    <Button android: id—"@+id/sendEmai1Btn"
android: layout width="wrap_content "android:
layout height—"wrap content" android: text="
Send Email" / > </LinearLayout>
```

OUTPUT:



RESULT:

Thus, the above experiment for designing the front-end of an Email Sending App using Android Studio has been successfully created and implemented.