Practical 28

Practical Related Questions

1.Explain validation of user input?

Input Validation eliminates the errors that can be done by user while giving inputs to our app. For example, if we want to get the user's email, we can check the entered email is a valid email or not before storing it inside the database. So let’s start our Android Form Validation tutorial.

2.List and explain various GUI components used to design the login form with validation.

TextView

EditText

Button

ImageButton

ToggleButton

RadioButton

RadioGroup

CheckBox

AutoCompleteTextView

ProgressBar

Spinner

TimePicker

DatePicker

SeekBar

AlertDialog

Switch

RatingBar

Practical 28

3.Differentiate between Text View and Edit Text View.

Edittext	Text view
EditText is used for user input.	TextView is used to display text and is not editable by the user. TextView can be updated programatically at any time.
EditText is Input Type/Field for inputting text	**TextView** is TextField for showing text

Exercise

MainActivity.java

```
package com.example.practical28;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.util.Patterns;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import java.util.regex.Pattern;

public class MainActivity extends AppCompatActivity {
    EditText textInputUsername,email,textInputPassword;
    Button login;
    int attempts=0;
    private static final Pattern PASSWORD_PATTERN =
            Pattern.compile("^" +
                    "(?=.*[0-9])" +
                    "(?=.*[a-z])" +
                    "(?=.*[A-Z])" +
                    "(?=.*[a-zA-Z])" +
                    "(?=.*[@#$%^&+=])" +
                    "(?=\\S+$)" +
```

Practical 28

```
                    ".{4,}" +
                    "$");
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        textInputUsername=(EditText) findViewById(R.id.username);
        textInputPassword=(EditText) findViewById(R.id.password);
        email=(EditText) findViewById(R.id.email);
        login=(Button) findViewById(R.id.login);

        login.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                attempts++;
                if (validateEmail() & validateUsername() & validatePassword()) {
                    Toast.makeText(MainActivity.this, "Login successful, attempt:
 "+attempts, Toast.LENGTH_SHORT).show();
                }
                else{
                    Toast.makeText(MainActivity.this, "Login Fail, attempt: "+att
empts, Toast.LENGTH_SHORT).show();

                }
            }
        });
    }
    private boolean validateEmail(){
        String emailInput=email.getText().toString().trim();
        if(emailInput.isEmpty()){
            email.setError("Field can't be empty");
            return false;
        }else if (!Patterns.EMAIL_ADDRESS.matcher(emailInput).matches()) {
            email.setError("Please enter a valid email address");
            return false;
        }else{
            email.setError(null);
            return true;
        }
    }

    private boolean validateUsername() {
        String usernameInput = textInputUsername.getText().toString().trim();
```

Practical 28

```
        if (usernameInput.isEmpty()) {
            textInputUsername.setError("Field can't be empty");
            return false;
        } else if (usernameInput.length() > 15) {
            textInputUsername.setError("Username too long");
            return false;
        } else {
            textInputUsername.setError(null);
            return true;
        }
    }

    private boolean validatePassword() {
        String passwordInput = textInputPassword.getText().toString().trim();

        if (passwordInput.isEmpty()) {
            textInputPassword.setError("Field can't be empty");
            return false;
        } else if (!PASSWORD_PATTERN.matcher(passwordInput).matches()) {
            textInputPassword.setError("Password too weak");
            return false;
        } else {
            textInputPassword.setError(null);
            return true;
        }
    }
}
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity"
    android:gravity="center"
    android:padding="30dp">
```

Practical 28

```
    <EditText
        android:id="@+id/username"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName"
        android:hint="username"
        android:layout_marginBottom="20dp"
        />
    <EditText
        android:id="@+id/email"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textEmailAddress"
        android:hint="Email"
        android:layout_marginBottom="20dp"/>
    <EditText
        android:id="@+id/password"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPassword"
        android:hint="Password"

        android:layout_marginBottom="20dp"/>
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Login"
        android:id="@+id/login"

        />
</LinearLayout>
```