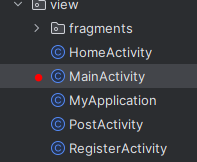
**ACTIVIDAD N°8 :**

**Clase Java de la vista: MainActivity**

1. MainActivity (login):



**MÉTODOS**

1. **onCreate(…)**

protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
  
 binding = ActivityMainBinding.inflate(getLayoutInflater());  
 setContentView(binding.getRoot());  
 viewModel = new ViewModelProvider(this, new MainViewModelFactory(this)).get(MainViewModel.class);  
 authProvider=new AuthProvider(this);  
 manejarEventos();  
}

**@Override**: Indica que el método onCreate está sobrescribiendo un método de la clase base Activity.

**protected void onCreate(Bundle savedInstanceState)** se llama cuando la actividad se está iniciando. Bundle savedInstanceState contiene el estado previamente guardado de la actividad, si existe.

**super.onCreate(savedInstanceState)** llama al método onCreate de la clase base Activity para realizar la configuración inicial de la actividad.

**binding = ActivityMainBinding.inflate(getLayoutInflater())** Utiliza el objeto LayoutInflater para crear una instancia de ActivityMainBinding, que es una clase generada automáticamente para la vinculación de vistas (View Binding).

**setContentView(binding.getRoot())** establece la vista raíz de la actividad a la vista inflada por el binding.

**viewModel = new ViewModelProvider(this, new MainViewModelFactory(this)).get(MainViewModel.class)** crea una instancia de ViewModelProvider para obtener el MainViewModel utilizando una fábrica personalizada MainViewModelFactory.

**authProvider = new AuthProvider(this)** crea una instancia de AuthProvider, que probablemente maneja la autenticación de usuarios.

**manejarEventos()** llama al método  manejarEventos, que configura los manejadores de eventos para la actividad.

**2. private void manejarEventos()**

Este método configura los eventos de clic para los botones de registro y login en la actividad principal

private void manejarEventos() {  
 binding.tvRegistro.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent intent = new Intent(MainActivity.this, RegisterActivity.class);  
 intent.setFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TASK | Intent.FLAG\_ACTIVITY\_NEW\_TASK);  
 startActivity(intent);  
 }  
 });  
 binding.btLogin.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 String email = binding.itUsuario.getText().toString().trim();  
 String pass = binding.itPassword.getText().toString().trim();  
  
 if (!Validaciones.validarMail(email)) {  
 showToast("Email incorrecto");  
 return;  
 }  
 if (!Validaciones.controlarPasword(pass)) {  
 showToast("Password incorrecto");  
 return;  
 }  
  
 viewModel.login(email, pass).observe(MainActivity.this, user\_id -> {  
 if (user\_id!=null) {  
  
 Intent intent=new Intent(MainActivity.this,HomeActivity.class);  
 // intent.putExtra("user\_id",user\_id);  
 startActivity(intent);  
 } else {  
 showToast("Login fallido");  
 }  
 });  
 }  
});  
}

**binding.tvRegistro.setOnClickListener** configura un OnClickListener para el TextView de registro (tvRegistro):

* Al hacer clic: Se crea un Intent para iniciar RegisterActivity.
* Flags del Intent: Intent.FLAG\_ACTIVITY\_CLEAR\_TASK y Intent.FLAG\_ACTIVITY\_NEW\_TASK aseguran que la nueva actividad sea la única en la pila de tareas.

**startActivity(intent)** Inicia RegisterActivity.

**binding.btLogin.setOnClickListener** configura un OnClickListener para el botón de login (btLogin) al hacer clic:

* Obtiene el correo electrónico y la contraseña ingresados por el usuario.
* Validación del correo electrónico: Utiliza Validaciones.validarMail(email) para verificar si el correo es válido. Si no lo es, muestra un mensaje de error y retorna.
* Validación de la contraseña: Utiliza Validaciones.controlarPasword(pass) para verificar si la contraseña es válida. Si no lo es, muestra un mensaje de error y retorna.
* Llamada al ViewModel: Llama al método login del viewModel con el correo y la contraseña.

**Observación del resultado del login**:

Si el login es exitoso (user\_id != null):

Crea un Intent para iniciar HomeActivity.

Inicia HomeActivity.

Si el login falla:

Muestra un mensaje de error indicando que el login falló.

**3. private void showToast(String message)**

private void showToast(String message) {  
 Toast.makeText(MainActivity.this, message, Toast.LENGTH\_LONG).show();  
}

**Toast.makeText**: Crea un Toast, que es una pequeña ventana emergente que muestra información al usuario.

**getApplicationContext()**: Obtiene el contexto de la aplicación, necesario para mostrar el Toast.

**message**: El mensaje que se mostrará en el Toast.

**Toast.LENGTH\_SHORT**: La duración del Toast, en este caso, breve.

Cuando se llama a showToast, se mostrará un mensaje breve en la pantalla

**4. protected void onResume()**

@Override  
protected void onResume() {  
 super.onResume();  
 limpiarCampos();

}

**protected void onResume()**: Sobrescribe el método onResume de la clase base Activity.

**super.onResume()**: Llama al método onResume de la clase base para asegurarse de que cualquier comportamiento predeterminado se ejecute correctamente.

**limpiarCampos()**: Llama a un método personalizado limpiarCampos, que probablemente se encarga de limpiar o restablecer los campos de entrada en la interfaz de usuario.

**5. public void limpiarCampos()**

Este método es para la desconexión de un usuario utilizando Parse.

private void limpiarCampos() {  
 binding.itUsuario.setText("");  
 binding.itPassword.setText("");  
  
 }

**CÓDIGO**

import android.content.Intent;  
import android.os.Bundle;  
import android.util.Log;  
import android.view.View;  
import android.widget.Toast;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.lifecycle.ViewModelProvider;  
import com.example.appchat.databinding.ActivityMainBinding;  
import com.example.appchat.providers.AuthProvider;  
import com.example.appchat.util.Validaciones;  
import com.example.appchat.viewmodel.MainViewModel;  
import com.example.appchat.viewmodel.MainViewModelFactory;  
  
public class MainActivity extends AppCompatActivity {  
 private ActivityMainBinding binding;  
 private MainViewModel viewModel;  
 private AuthProvider authProvider;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
  
 binding = ActivityMainBinding.inflate(getLayoutInflater());  
 setContentView(binding.getRoot());  
 viewModel = new ViewModelProvider(this, new MainViewModelFactory(this)).get(MainViewModel.class);  
 authProvider=new AuthProvider(this);  
 manejarEventos();  
 }  
  
 @Override  
 protected void onStart() {  
 super.onStart();  
   
  
  
 private void manejarEventos() {  
 binding.tvRegistro.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent intent = new Intent(MainActivity.this, RegisterActivity.class);  
 intent.setFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TASK | Intent.FLAG\_ACTIVITY\_NEW\_TASK);  
 startActivity(intent);  
 }  
 });  
 binding.btLogin.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 String email = binding.itUsuario.getText().toString().trim();  
 String pass = binding.itPassword.getText().toString().trim();  
  
 if (!Validaciones.validarMail(email)) {  
 showToast("Email incorrecto");  
 return;  
 }  
 if (!Validaciones.controlarPasword(pass)) {  
 showToast("Password incorrecto");  
 return;  
 }  
  
 viewModel.login(email, pass).observe(MainActivity.this, user\_id -> {  
 if (user\_id!=null) {  
  
 Intent intent=new Intent(MainActivity.this,HomeActivity.class);  
 // intent.putExtra("user\_id",user\_id);  
 startActivity(intent);  
 } else {  
 showToast("Login fallido");  
 }  
 });  
 }  
 });  
 }  
  
 private void showToast(String message) {  
 Toast.makeText(MainActivity.this, message, Toast.LENGTH\_LONG).show();  
 }  
 @Override  
 protected void onResume() {  
 super.onResume();  
 limpiarCampos();  
 }  
 private void limpiarCampos() {  
 binding.itUsuario.setText("");  
 binding.itPassword.setText("");  
  
 }  
  
  
}