**Detalhes sobre Projeto Chat com Fitebase com Java**

Arquivo principal “**AndroidManifet.xml**” do diretório “**manifest**”

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.ramerips\_cursos">  
  
 <application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/AppTheme">  
  
 <!-- ###- TECLAS DE ATALHOS PARA LOCALIZAR CLASSES OU ARQUIVOS.-### -->  
 <!-- ##- Obs: CTRL+N -> Localiza uma "Classe", CTRL+SHIFT+N -> Localiza um "Arquivo" e CTRL+SHIFT+F -> Localiza um "Caminho" -## -->  
 <!-- ####- Aqui abaixo podemos alterar o carregamento da tela principal. -#### -->  
 <!-- Quando se roda uma aplicação, abrirá a primeira da sequência, no caso, logo abaixo. -->  
 <!-- Aqui logo abaixo será o primeiro layout da aplicação à abrir quando startado. -->  
  
 <!--Aqui abaixo temos a activity "LoginActivity" criada para esta aplicação -->  
 <activity android:name=".LoginActivity">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
  
  
 <!--Aqui abaixo temos a activity "ErroActivity" criada para esta aplicação -->  
 <activity android:label="Página de Erro Número 404!" android:name=".erros.ErroActivity">  
  
 </activity>  
  
  
 <!--Aqui abaixo temos a activity "ContatosActivity" criada para esta aplicação -->  
 <activity android:label="Contatos" android:name=".ContatosActivity">  
  
 </activity>  
  
 <!--Aqui abaixo temos a activity "MessagesActivity" criada para esta aplicação -->  
 <activity android:name=".MessagesActivity">  
  
 </activity>  
  
 <!--Aqui abaixo temos a activity "RegistrarActivity" criada para esta aplicação -->  
 <activity android:name=".RegistrarActivity">  
  
 </activity>  
  
 </application>  
  
</manifest>

Arquivo classe “**ContatosActivity.java**” do diretório “**java**”

package com.ramerips\_cursos;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.os.Bundle;  
import android.util.Log;  
  
import com.google.firebase.firestore.DocumentSnapshot;  
import com.google.firebase.firestore.EventListener;  
import com.google.firebase.firestore.FirebaseFirestore;  
import com.google.firebase.firestore.FirebaseFirestoreException;  
import com.google.firebase.firestore.QuerySnapshot;  
  
import java.util.List;  
  
import javax.annotation.Nullable;  
  
public class ContatosActivity extends AppCompatActivity {  
  
 //  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*act\_contatos*);  
  
 //Aqui é chamado o método "fetchUsers()", responsável por resgatar ou buscar os usuário no "Firebase"  
 fetchUsers();  
 }  
  
 //Este método será o responsável por resgatar ou buscar o usuário requisitado no "Firebase"  
 private void fetchUsers(){  
 //Aqui é verificado, encontrado e buscado junto ao "Firebase" a coleção "users" e seu conteúdo  
 FirebaseFirestore.*getInstance*().collection("/users")  
 //Aqui é adicionado instantaneamente em uma lista por meio do método "EventListener<QuerySnapshot>()"  
 .addSnapshotListener(new EventListener<QuerySnapshot>() {  
 //Aqui abaixo é um evento de escuta da execução da atividade ocorrida por meio do método "onEvent()" e seus parâmetros  
 @Override  
 public void onEvent(@Nullable QuerySnapshot queryDocumentSnapshots, @Nullable FirebaseFirestoreException e) {  
 //Aqui abaixo será verificado se a exceção(Erros) -> "e" for diferente de nulo  
 if(e != null){  
 //Aqui abaixo temos uma mensagem "Teste" em modo "Log.e" com a impressão do conteúdo de erro  
 //que é mostada ao desenvolvedor no painel do "6: Logcat"  
 Log.*e*("Teste", e.getMessage(), e);  
 //Aqui temos o retorno da execução em questão  
 return;  
 }  
 //Aqui temos o objeto "objDocuments" do tipo "List<DocumentSnapshot>" que receberá todos os dados ou conteúdo  
 //de documentos contidos dentro da consulta instantânia "queryDocumentSnapshots" e que é trazida pelo método  
 //"getDocuments()", onde consta cada elemento ou usuário trazido pelo mesmo  
 List<DocumentSnapshot> objDocuments = queryDocumentSnapshots.getDocuments();  
  
 //  
 for (DocumentSnapshot objDocConteudo: objDocuments){  
 //  
 User user = objDocConteudo.toObject(User.class);  
 //  
 Log.*d*("Teste", user.getUsername());  
 }  
 }  
 });  
  
 }  
}

Arquivo classe “**LoginActivity.java**” da aplicação do diretório “**java**”

package com.ramerips\_cursos;  
  
//<!--###- TECLAS DE ATALHOS PARA LOCALIZAR CLASSES OU ARQUIVOS.-###-->  
//<!--##- Obs: CTRL+N -> Localiza uma "Classe", CTRL+SHIFT+N -> Localiza um "Arquivo" e CTRL+SHIFT+F -> Localiza um "Caminho" -##-->  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.util.Log;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import com.google.android.gms.tasks.OnCompleteListener;  
import com.google.android.gms.tasks.OnFailureListener;  
import com.google.android.gms.tasks.Task;  
import com.google.firebase.auth.AuthResult;  
import com.google.firebase.auth.FirebaseAuth;  
  
import static android.widget.Toast.*LENGTH\_LONG*;  
  
public class LoginActivity extends AppCompatActivity {  
  
 //Aqui serão criados os atributos relacionados aos campos do layout "act\_login"  
 private EditText mEditText\_Email;  
 private EditText mEditText\_Senha;  
 private Button mBtn\_Entrar;  
 private TextView mtextView\_CriarConta;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*act\_login*);  
  
 //Aqui serão recuperados os dados inseridos nos campos do layout "act\_login"  
 mEditText\_Email = findViewById(R.id.*editText\_Email*);  
 mEditText\_Senha = findViewById(R.id.*editText\_Senha*);  
 mBtn\_Entrar = findViewById(R.id.*btn\_Cadastrar*);  
 mtextView\_CriarConta = findViewById(R.id.*textView\_CriarConta*);  
  
 //Aqui será criado um evento de clique acionado pelo botão "btn\_Entrar" do layout "act\_login.xml"  
 mBtn\_Entrar.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 //Aqui abaixo as variáveis do tipo string(email e senha), receberão os valores contidos nos dados recuperados  
 //por meio do método "getText()" e transformados em string por meio do método "toString()"  
 String email = mEditText\_Email.getText().toString();  
 String senha = mEditText\_Senha.getText().toString();  
  
 //Aqui serão impressas as variáveis por meio do "Log.i" e as strings "Teste" que as antecede -> (Aqui é teste do desenvolvedor)  
 Log.*i*("Teste", email);  
 Log.*i*("Teste", senha);  
  
 //Aqui será verificado se os campos do formulário do email e senha estão inválidos ou nulos(Vazios), onde a mensagem do "Toast"  
 //só será impressa para o usuário caso estejam com problemas "Inválidos ou vazios".  
 if (email == null || email.isEmpty() || senha == null || senha.isEmpty()){  
 //Aqui será implementado um estilo de mensagem volátil do tipo "Toast" que será exibida ao usuário, onde existe a possibilidade de  
 //ser do tipo "Toast.LENGTH\_LONG(Longa duração) e Toast.LENGTH\_SHORT(Rápida duração)", onde desaparece automaticamente.  
 Toast.*makeText*(LoginActivity.this, "E-mail ou senha devem ser preenchidos corretamente", *LENGTH\_LONG*).show();  
 return;  
 }  
  
 //Aqui abaixo é o evento que escutará o sucesso da execução da tarefa caso insira um novo usuário  
 FirebaseAuth.*getInstance*().signInWithEmailAndPassword(email, senha)  
 .addOnCompleteListener(new OnCompleteListener<AuthResult>() {  
 @Override  
 public void onComplete(@NonNull Task<AuthResult> task) {  
 //Aqui abaixo temos uma mensagem "Teste" em modo "Log.i" com a impressão da "usuário" em formato de string  
 //que é mostada ao desenvolvedor no painel do "6: Logcat" com o link de acesso da mesma  
 Log.*i*("Teste", task.getResult().getUser().getUid());  
  
 //Aqui é criado um novo objeto "objIntent" do tipo "Intent" que receberá uma nova página ou layout  
 //"MessagesActivity" e determinada a sua abertura pela "LoginActivity"  
 Intent objIntent = new Intent(LoginActivity.this, MessagesActivity.class);  
 //Aqui logo abaixo o objeto "objIntent" setará o método "setFlags()" que determina que a página  
 //ou layout a ser aberta seja a do objeto "objIntent" que é a "MessagesActivity"  
 objIntent.setFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TASK* | Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  
 //Aqui temos a inicialização da activity "MessagesActivity" por meio do método "startActivity()"  
 startActivity(objIntent);  
 }  
 })  
 //Aqui abaixo é o evento que escutará o erro da execução da tarefa, caso não insira um novo usuário  
 .addOnFailureListener(new OnFailureListener() {  
 @Override  
 public void onFailure(@NonNull Exception e) {  
 //Aqui abaixo temos uma mensagem "Teste" em modo "Log.i" com a impressão da "usuário" em formato de string  
 //que é mostada ao desenvolvedor no painel do "6: Logcat" com o link de acesso da mesma  
 Log.*i*("Teste", e.getMessage());  
 }  
 });  
 }  
 });  
  
 //Aqui será criado um evento de clique acionado pelo TextView "TextView\_CriarConta" do layout "act\_login"  
 mtextView\_CriarConta.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 //Aqui o objeto "intent" receberá uma nova "activity" para ser startada ou iniciada em um outro layout  
 //que é a página "RegistrarActivity" que irá ser aberta pelo evento de clique  
 Intent intent = new Intent(LoginActivity.this, RegistrarActivity.class);  
 //Aqui será startada a activity "RegistrarActivity" recebida pelo objeto "intent"  
 startActivity(intent);  
 }  
 });  
 }  
}

Arquivo classe “**MessagesActivity.java**” do diretório “**java**”

package com.ramerips\_cursos;  
  
//<!--###- TECLAS DE ATALHOS PARA LOCALIZAR CLASSES OU ARQUIVOS.-###-->  
//<!--##- Obs: CTRL+N -> Localiza uma "Classe", CTRL+SHIFT+N -> Localiza um "Arquivo" e CTRL+SHIFT+F -> Localiza um "Caminho" -##-->  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.Menu;  
import android.view.MenuItem;  
  
import com.google.firebase.auth.FirebaseAuth;  
  
public class MessagesActivity extends AppCompatActivity {  
  
 //  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*act\_messages*);  
  
 //Aqui será chamado o método de verificação "verifyAuthentication()"  
 verifyAuthentication();  
 }  
  
 //Este método verificará se o usuário está logado ou não junto ao "Firebase"  
 private void verifyAuthentication(){  
 //Aqui verifica se o usuário é nulo ou não...  
 if(FirebaseAuth.*getInstance*().getUid() == null){  
 //Aqui é criado um novo objeto "objIntent" do tipo "Intent" que receberá uma nova página ou layout  
 //"LoginActivity" e determinada a sua abertura pela "MessagesActivity", caso o usuário não esteja  
 //devidamente logado.  
 Intent objIntent = new Intent(MessagesActivity.this, LoginActivity.class);  
 //Aqui logo abaixo o objeto "objIntent" setará o método "setFlags()" que determina que a página  
 //ou layout a ser aberta seja a do objeto "objIntent" que é a "MessagesActivity"  
 objIntent.setFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TASK* | Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  
 //Aqui temos a inicialização da activity "MessagesActivity" por meio do método "startActivity()"  
 startActivity(objIntent);  
 }  
 }  
  
 //Aqui será implementado o método "onCreateOptionsMenu()" que receberá como parâmetro a classe "Menu" e o  
 //arquivo "menu.xml"  
 public boolean onCreateOptionsMenu(Menu menu){  
 //Aqui o método "getMenuInflater()", fará com que o arquivo de menu apareça ou seja acionado em determinada  
 //pagina ou layoute mostrando assim suas opçoes para o usuário  
 getMenuInflater().inflate(R.menu.*menu*, menu);  
 //Aqui daremos "true" ou verdadeiro para o seu retorno  
 return true;  
 }  
  
 //Aqui abaixo será implementado um método de escuta dos eventos de clique destes menus quando clicado  
 public boolean onOptionsItemSelected(MenuItem item) {  
 //Aqui abaixo será implementado com o "suitch(Escolha)" uma maneira de identifica qual foi o item ou "id" do menu que foi  
 //clicado em uma das opções de menu existe neste "switch(Escolha)"  
 switch (item.getItemId()) {  
 //Aqui é informado que este item abaixo é o "id" contatos  
 case R.id.*idContatos*:  
  
 //Aqui é criado um novo objeto "objIntent" do tipo "Intent" que receberá uma nova página ou layout  
 //"ContatosActivity" e determinada a sua abertura pela "MessagesActivity" caso o menu "idContatos" seja clicado  
 Intent objIntent = new Intent(MessagesActivity.this, ContatosActivity.class);  
 //Aqui logo abaixo o objeto "objIntent" setará o método "setFlags()" que determina que a página  
 //ou layout a ser aberta seja a do objeto "objIntent" que é a "MessagesActivity"  
 objIntent.setFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TASK* | Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  
 //Aqui temos a inicialização da activity "MessagesActivity" por meio do método "startActivity()"  
 startActivity(objIntent);  
  
 //Aqui abaixo é dado um encerramento de escolha das opções deste "switch"  
 break;  
  
  
 //Aqui é informado que este item abaixo é o "id" sair  
 case R.id.*idSair*:  
 //Aqui abaixo será identificado qual é o usuário que está logado e irá deslogar este usuário com o método signOut()  
 FirebaseAuth.*getInstance*().signOut();  
 //Aqui abaixo com o método "verifyAuthentication()" será verificado se realmente o usuário não está logado  
 verifyAuthentication();  
 //Aqui abaixo é dado um encerramento de escolha das opções deste "switch"  
 break;  
 }  
 //Aqui...  
 return super.onOptionsItemSelected(item);  
 }  
}

Arquivo classe “**RegistrarActivity.java**” do diretório “**java**”

package com.ramerips\_cursos;  
  
//<!--###- TECLAS DE ATALHOS PARA LOCALIZAR CLASSES OU ARQUIVOS.-###-->  
//<!--##- Obs: CTRL+N -> Localiza uma "Classe", CTRL+SHIFT+N -> Localiza um "Arquivo" e CTRL+SHIFT+F -> Localiza um "Caminho" -##-->  
  
import androidx.annotation.NonNull;  
import androidx.annotation.Nullable;  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.graphics.Bitmap;  
import android.graphics.drawable.BitmapDrawable;  
import android.net.Uri;  
import android.os.Bundle;  
import android.provider.MediaStore;  
import android.util.Log;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ImageView;  
import android.widget.Toast;  
  
import com.google.android.gms.tasks.OnCompleteListener;  
import com.google.android.gms.tasks.OnFailureListener;  
import com.google.android.gms.tasks.OnSuccessListener;  
import com.google.android.gms.tasks.Task;  
import com.google.firebase.auth.AuthResult;  
import com.google.firebase.auth.FirebaseAuth;  
import com.google.firebase.firestore.DocumentReference;  
import com.google.firebase.firestore.FirebaseFirestore;  
import com.google.firebase.storage.FirebaseStorage;  
import com.google.firebase.storage.StorageReference;  
import com.google.firebase.storage.UploadTask;  
  
import java.io.IOException;  
import java.util.UUID;  
  
import static android.content.Intent.*ACTION\_PICK*;  
  
public class RegistrarActivity extends AppCompatActivity {  
  
 //Aqui serão criados os atributos relacionados aos campos do layout "act\_login"  
 private EditText mEditText\_Nome;  
 private EditText mEditText\_Email;  
 private EditText mEditText\_Senha;  
 private Button mBtn\_Cadastrar;  
 private Button mBtn\_FotoUsuario;  
 private ImageView mImg\_Foto;  
 private Uri mSelectedUri;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*act\_registrar*);  
  
 //Aqui serão recuperados os dados inseridos nos campos do layout "act\_registrar.xml"  
 mEditText\_Nome = findViewById(R.id.*editText\_Nome*);  
 mEditText\_Email = findViewById(R.id.*editText\_Email*);  
 mEditText\_Senha = findViewById(R.id.*editText\_Senha*);  
 mBtn\_Cadastrar = findViewById(R.id.*btn\_Cadastrar*);  
 mBtn\_FotoUsuario = findViewById(R.id.*idBtnFotoUsuario*);  
 mImg\_Foto = findViewById(R.id.*idImg\_Foto*);  
  
 //Aqui será criado um evento de clique acionado pelo botão "btn\_Cadastrar" do layout "act\_registrar.xml"  
 mBtn\_FotoUsuario.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 //Aqui chamamos o método ou função "selecionarFoto()", que seleciona a foto desejada para o objeto "mBtn\_FotoUsuario"  
 selecionarFoto();  
 }  
 });  
  
 //Aqui será criado um evento de clique acionado pelo botão "btn\_Cadastrar" do layout "act\_registrar.xml"  
 mBtn\_Cadastrar.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 //Aqui chamamos o método ou função "createUser()", ou "criar usuário" para efetuar a sua execução  
 createUser();  
 }  
 });  
 }  
  
 //Aqui será implementado uma codificação que retornará um resultado em código de uma outra activity, no caso galeria  
 @Override  
 protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {  
 super.onActivityResult(requestCode, resultCode, data);  
 //Aqui será verificado se o "requestCode" é igual a "0" para poder executar o conteúdo do "if"  
 if(requestCode == 0){  
 //Aqui abaixo a variável "mSelectedUri" receberá os dados do "data" por meio do método "getData()"  
 mSelectedUri = data.getData();  
  
 //Aqui abaixo será criado um objeto "objBitmap" do tipo "Bitmap" e que inicialmente receberá um valor nulo "null"  
 Bitmap objBitmap = null;  
  
 //Aqui abaixo temos a implementação do "try e catch", onde será capturada as excessões  
 try {  
 //Aqui o objeto "objBitmap" receberá todos os parâmetros necessário de sua implementação  
 objBitmap = MediaStore.Images.Media.*getBitmap*(getContentResolver(), mSelectedUri);  
 //Aqui o objeto "mImg\_Foto" setará o método "setImageDrawable()" onde está a imagem, pegando-a  
 mImg\_Foto.setImageDrawable(new BitmapDrawable(objBitmap));  
 //Aqui o objeto "mBtn\_FotoUsuario" setará o método "setAlpha(0)" de valor "0", ou seja, a imagem "mBtn\_FotoUsuario"  
 //irá desaparecer e sugirá a imagem "mImg\_Foto"  
 mBtn\_FotoUsuario.setAlpha(0);  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
 //Aqui será implementado o método de selecionar uma foto na galeria do sistema ou máquina  
 private void selecionarFoto(){  
 //Aqui será criada uma nova "Intent(Intenção)" de "ACTION\_PICK(Ação de se obter algo)", no caso seria uma imagem na galeria  
 Intent objIntent = new Intent(Intent.*ACTION\_PICK*);  
 //Aqui o objeto "objIntent" setará o método "setType()" para selecionar ou buscar o tipo de dados, no caso, todos os tipos de imagens  
 objIntent.setType("image/\*");  
 //Aqui será passado como parâmetro o objeto intent "objIntent" criado e o código recebido pelo método "onActivityResult()"  
 startActivityForResult(objIntent, 0);  
 }  
  
 //Aqui temos a implementação de um método "createUser()", ou seja, uma função de criar um usuário novo  
 private void createUser() {  
 //Aqui abaixo as variáveis do tipo string(email e senha), receberão os valores contidos nos dados recuperados  
 //por meio do método "getText()" e transformados em string por meio do método "toString()"  
 String nome = mEditText\_Nome.getText().toString();  
 String email = mEditText\_Email.getText().toString();  
 String senha = mEditText\_Senha.getText().toString();  
  
 //Aqui será verificado se os campos do formulário da senha e email estão inválidos ou nulos(Vazios), onde a mensagem do "Toast"  
 //só será impressa para o usuário caso estejam com problemas "Inválidos ou vazios".  
 if (nome == null || nome.isEmpty() || email == null || email.isEmpty() || senha == null || senha.isEmpty()){  
 //Aqui será implementado um estilo de mensagem volátil do tipo "Toast" que será exibida ao usuário, onde existe a possibilidade de  
 //ser do tipo "Toast.LENGTH\_LONG(Longa duração) e Toast.LENGTH\_SHORT(Rápida duração)", onde desaparece automaticamente.  
 Toast.*makeText*(this, "Nome, senha e email devem ser preenchidos corretamente", Toast.*LENGTH\_LONG*).show();  
 return;  
 }  
  
 //Aqui será implementada a criação de um "Id" de usuário criptografados e que receberá um email e uma senha por meio do "FirebaseAuth"  
 FirebaseAuth.*getInstance*().createUserWithEmailAndPassword(email, senha)  
 //Aqui será inserido no "Firebase" na aba "Authentication", um "email" e uma "senha" e também criado um "Id" de usuário criptografado  
 .addOnCompleteListener(new OnCompleteListener<AuthResult>(){  
 @Override  
 public void onComplete(@NonNull Task<AuthResult> task) {  
 //Aqui será verificado um possível sucesso na executada da "task(Tarefa)" por meio da autenticação junto ao "Firebase"  
 //caso sucesso, será impresso o resultado desta tarefa executada por meio dos métodos "getResult(), getUser() e getUid()".  
 if(task.isSuccessful()) {  
 //Aqui temos uma mensagem em modo "Log.i" com a impressão da tarefa que é mostada ao desenvolvedor no painel do  
 //"6: Logcat" com o conteúdo logo abaixo que cria um "id" criptografado para o usuário  
 Log.*i*("Teste", task.getResult().getUser().getUid()); //Aqui é apenas para teste do desenvolvedor  
  
 //Aqui será chamado o método "salvarUsuarioNoFirebase()"  
 salvarUsuarioNoFirebase();  
 }  
 //Obs:Verificar junto ao "Firebase" na aba "Authentication" a inclusão do "Email" e a criação do "Id" do usuário  
 //criptografado.  
  
 //Obs:  
 //1->The email address is badly formatted. -> O endereço de email está mal formatado.  
 //2->The given password is invalid. [ Password should be at least 6 characters ] -> A senha fornecida  
 //é inválida. [A senha deve ter pelo menos 6 caracteres]  
 //3->A network error (such as timeout, interrupted connection or unreachable host) has occurred. -> Ocorreu  
 //um erro de rede (como tempo limite, conexão interrompida ou host inacessível).  
 //4->The email address is already in use by another account. -> O endereço de email já está sendo usado por outra conta.  
 }  
 })  
 //Aqui será adicionada ou capturada uma falha ou erro de "Exception" na execução desta autenticação ou "task(Tarefa)"  
 .addOnFailureListener(new OnFailureListener() {  
 @Override  
 public void onFailure(@NonNull Exception e) {  
 //Aqui temos uma mensagem de erro em modo "Log.i" que é mostada ao desenvolvedor no painel do "6: Logcat"  
 //com o conteúdo logo abaixo ou mensagem de erro específica  
 Log.*i*("Teste", e.getMessage()); //Aqui é apenas para teste do desenvolvedor  
  
 //Obs: Tradução de Inglês para Português  
 //1->The email address is badly formatted. -> O endereço de email está mal formatado.  
 //2->The given password is invalid. [ Password should be at least 6 characters ] -> A senha fornecida  
 //é inválida. [A senha deve ter pelo menos 6 caracteres]  
 //3->A network error (such as timeout, interrupted connection or unreachable host) has occurred. -> Ocorreu  
 //um erro de rede (como tempo limite, conexão interrompida ou host inacessível).  
 //4->The email address is already in use by another account. -> O endereço de email já está sendo usado por outra conta.  
 }  
 });  
 }  
  
 //Aqui será implementado um método que salvará os dados de um usuário cadastrado no "Firebase" mais a imagem de referência  
 private void salvarUsuarioNoFirebase() {  
 //Aqui abaixo temos uma variável "nomeArquivo" que receberá um código randômico(Aleiatório) em formato de string, porém  
 //um "Hash(Confusão, misturado)", ou seja, com os caracteres(Alfanuméricos) e todos eles misturados que identificará um usuário  
 String nomeArquivo = UUID.*randomUUID*().toString();  
 //Aqui abaixo o objeto "objReference" do tipo "StorageReference" e que não muda(final), receberá uma imagem e nome do arquivo  
 final StorageReference objReference = FirebaseStorage.*getInstance*().getReference("/images/" + nomeArquivo);  
 //Com o objeto "objReference" pronto, a imagem será enviada para o "Storage" do Firebase por meio do método "putFile()" e seu  
 //parâmetro "mSelectedUri" do tipo "Uri", pois essa imagem será identificada por uma "Hash" aleatória alfanumerica  
 objReference.putFile(mSelectedUri)  
 //Aqui com o método "addOnSuccessListener()" verifica se realmente houve a transferência da cópia da imagem selecionada  
 //na galeria para o "Storage" do Firebase com sucesso  
 .addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {  
 @Override  
 //Esse método abaixo entra automaticamente ao digitar o "new OnSuccessListener... " logo acima.  
 public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {  
 //Aqui o objeto "objReference" setará o método "getDownloadUrl()" para baixar a imagem que foi enviada ao "Firebase",  
 //por meio da "URL" criada, pois com essa referência você pode visualizar esta imagem que está no "Storage" do Firebase  
 //em qualquer navegador.  
 objReference.getDownloadUrl().addOnSuccessListener(new OnSuccessListener<Uri>() {  
 @Override  
 //Esse método abaixo entra automaticamente, trazendo a referência da "URL".  
 public void onSuccess(Uri uri) {  
 //Aqui abaixo temos uma mensagem "Teste" em modo "Log.i" com a impressão da "URL" criada pelo "Storage e o Firebase"  
 //que é mostada ao desenvolvedor no painel do "6: Logcat" com o link de acesso da mesma  
 Log.*i*("Teste", uri.toString());  
  
 //Aqui temos um objeto "uid"(usuário-id) que receberá toda a autenticação da instância do usuário  
 //criado no "Firebase do Google" em sua coleção do  
 String uid = FirebaseAuth.*getInstance*().getUid();  
 //Aqui abaixo a variável "username" receberá o valor do "mEditText\_Nome" em formato de string  
 String username = mEditText\_Nome.getText().toString();  
 //Aqui abaixo a variável "profileUrl" receberá o "uri" em formato de string  
 String profileUrl = uri.toString();  
  
 //Aqui será criado um novo  
 User user = new User(uid, username, profileUrl);  
  
 //Aqui abaixo é o evento que escutará o sucesso da execução da tarefa caso insira um novo usuário  
 FirebaseFirestore.*getInstance*().collection("users")  
 .add(user)  
 .addOnSuccessListener(new OnSuccessListener<DocumentReference>() {  
 @Override  
 public void onSuccess(DocumentReference documentReference) {  
 //Aqui abaixo temos uma mensagem "Teste" em modo "Log.i" com a impressão da "usuário" em formato de string  
 //que é mostada ao desenvolvedor no painel do "6: Logcat" com o link de acesso da mesma  
 Log.*i*("Teste", documentReference.getId());  
  
 //Aqui é criado um novo objeto "objIntent" do tipo "Intent" que receberá uma nova página ou layout  
 //"MessagesActivity" e determinada a sua abertura pela "RegistrarActivity"  
 Intent objIntent = new Intent(RegistrarActivity.this, MessagesActivity.class);  
 //Aqui logo abaixo o objeto "objIntent" setará o método "setFlags()" que determina que a página  
 //ou layout a ser aberta seja a do objeto "objIntent" que é a "MessagesActivity"  
 objIntent.setFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TASK* | Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  
 //Aqui temos a inicialização da activity "MessagesActivity" por meio do método "startActivity()"  
 startActivity(objIntent);  
  
 }  
 })  
 //Aqui abaixo é o evento que escutará o erro da execução da tarefa, caso não insira um novo usuário  
 .addOnFailureListener(new OnFailureListener() {  
 @Override  
 public void onFailure(@NonNull Exception e) {  
 //Aqui abaixo temos uma mensagem "Teste" em modo "Log.i" com a impressão da "erro" em formato  
 //de string que é mostada ao desenvolvedor no painel do "6: Logcat" com o link de acesso da mesma  
 Log.*i*("Teste", e.getMessage());  
 }  
 });  
 }  
 });  
 }  
 })  
 //Aqui será adicionada ou capturada uma falha ou erro de "Exception" na execução desta autenticação ou "task(Tarefa)"  
 .addOnFailureListener(new OnFailureListener() {  
 @Override  
 public void onFailure(@NonNull Exception e) {  
 //Aqui temos uma mensagem de erro em modo "Log.i" que é mostada ao desenvolvedor no painel do "6: Logcat"  
 //com o conteúdo logo abaixo ou mensagem de erro específica  
 Log.*i*("Teste", e.getMessage(), e);  
 }  
 });  
 }  
}

Arquivo classe “**User.java**” do diretório “**java**”

package com.ramerips\_cursos;  
  
//Aqui temos a implementação da classe usuário "User"  
public class User {  
  
 //Aqui teremos os atributos ou variáveis necessárias para a aplicação  
 private String uuid;  
 private String username;  
 private String profileUrl;  
  
 //Aqui temos um construtor padrão para viabilizar a execução  
 public User(){  
  
 }  
  
 //Aqui nós temos os atributos recebendo os valores de seus parâmetro do método abaixo  
 public User(String uuid, String username, String profileUrl) {  
 this.uuid = uuid;  
 this.username = username;  
 this.profileUrl = profileUrl;  
 }  
  
 //Aqui temos os "Gets e Sets" da classe "User"  
 public String getUuid() {  
 return uuid;  
 }  
 public String getUsername() {  
 return username;  
 }  
 public String getProfileUrl() {  
 return profileUrl;  
 }  
}

Arquivo classe “**act\_contatos.xml**” do diretório “**layout**”

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout  
 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"  
 tools:context=".ContatosActivity">  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:layout\_marginStart="2dp"  
 android:layout\_marginLeft="2dp"  
 android:layout\_marginTop="2dp"  
 android:layout\_marginEnd="2dp"  
 android:layout\_marginRight="2dp"  
 android:layout\_marginBottom="2dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.0"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.0" />  
  
</androidx.constraintlayout.widget.ConstraintLayout>

Arquivo layout “**act\_login.xml**” da aplicação do diretório “**layout**”

<?xml version="1.0" encoding="utf-8"?>  
<!--  
Aqui abaixo foi inserido a configuração "bgColor" de uma nova cor para a aplicação  
-->  
<androidx.constraintlayout.widget.ConstraintLayout  
 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:background="@color/bgColor"  
 tools:context=".LoginActivity">  
  
 <!-- Aqui é o "Hello World!" inicial da aplicação -->  
 <!--  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Hello World!"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
 -->  
  
 <!--  
 Aqui abaixo com o "bg\_edit\_text\_rounded" está sendo incluido o arquivo do diretório  
 "drawable" que configura esta formatação arredondada para a aplicação.  
 -->  
 <EditText  
 android:id="@+id/editText\_Email"  
 android:layout\_width="0dp"  
 android:layout\_height="50dp"  
 android:layout\_marginStart="16dp"  
 android:layout\_marginLeft="16dp"  
  
 android:layout\_marginTop="8dp"  
 android:layout\_marginEnd="16dp"  
 android:layout\_marginRight="16dp"  
 android:background="@drawable/bg\_edit\_text\_rounded"  
 android:ems="10"  
 android:hint="Email"  
 android:inputType="textEmailAddress"  
 android:paddingLeft="10dp"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/imageView" />  
  
 <EditText  
 android:id="@+id/editText\_Senha"  
 android:layout\_width="0dp"  
 android:layout\_height="50dp"  
 android:layout\_marginTop="8dp"  
 android:background="@drawable/bg\_edit\_text\_rounded"  
 android:paddingLeft="10dp"  
 android:hint="Senha"  
 android:ems="10"  
 android:inputType="textPassword"  
 android:textSize="18sp"  
 app:layout\_constraintEnd\_toEndOf="@+id/editText\_Email"  
 app:layout\_constraintStart\_toStartOf="@+id/editText\_Email"  
 app:layout\_constraintTop\_toBottomOf="@+id/editText\_Email" />  
  
 <Button  
 android:id="@+id/btn\_Cadastrar"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginStart="8dp"  
 android:layout\_marginLeft="8dp"  
 android:layout\_marginTop="24dp"  
 android:layout\_marginEnd="8dp"  
 android:layout\_marginRight="8dp"  
 android:background="@drawable/bg\_edit\_text\_rounded"  
 android:backgroundTint="@android:color/holo\_green\_dark"  
 android:text="@string/enter"  
 android:textColor="@android:color/white"  
 android:textSize="24sp"  
 android:textStyle="bold"  
 app:layout\_constraintEnd\_toEndOf="@+id/editText\_Senha"  
 app:layout\_constraintStart\_toStartOf="@+id/editText\_Senha"  
 app:layout\_constraintTop\_toBottomOf="@+id/editText\_Senha" />  
  
 <TextView  
 android:id="@+id/textView\_CriarConta"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="20dp"  
 android:text="@string/signIn"  
 android:textAlignment="center"  
 android:textColor="@android:color/holo\_green\_dark"  
 android:textSize="16sp"  
 android:textStyle="bold"  
 app:layout\_constraintEnd\_toEndOf="@+id/btn\_Cadastrar"  
 app:layout\_constraintStart\_toStartOf="@+id/btn\_Cadastrar"  
 app:layout\_constraintTop\_toBottomOf="@+id/btn\_Cadastrar" />  
  
 <ImageView  
 android:id="@+id/imageView"  
 android:layout\_width="250dp"  
 android:layout\_height="150dp"  
 android:layout\_marginTop="8dp"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:srcCompat="@drawable/cadastro\_cliente" />  
  
</androidx.constraintlayout.widget.ConstraintLayout>

Arquivo layout “**act\_messages.xml**” da aplicação do diretório “**layout**”

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout  
 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"  
 tools:context=".MessagesActivity">  
  
</androidx.constraintlayout.widget.ConstraintLayout>

Arquivo layout “**act\_registrar.xml**” da aplicação do diretório “**layout**”

<?xml version="1.0" encoding="utf-8"?>  
<!--  
Aqui abaixo foi inserido a configuração "bgColor" de uma nova cor para a aplicação  
-->  
<androidx.constraintlayout.widget.ConstraintLayout  
 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:background="@color/bgColor"  
 tools:context=".LoginActivity">  
  
 <!-- Aqui é o "Hello World!" inicial da aplicação -->  
 <!--  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Hello World!"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
 -->  
  
 <EditText  
 android:id="@+id/editText\_Nome"  
 android:layout\_width="0dp"  
 android:layout\_height="50dp"  
 android:layout\_marginStart="16dp"  
 android:layout\_marginLeft="16dp"  
 android:layout\_marginTop="8dp"  
 android:layout\_marginEnd="16dp"  
 android:layout\_marginRight="16dp"  
 android:background="@drawable/bg\_edit\_text\_rounded"  
 android:ems="10"  
 android:hint="Nome"  
 android:inputType="textPersonName"  
 android:paddingLeft="10dp"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/idBtnFotoUsuario" />  
  
 <!--  
 Aqui abaixo com o "bg\_edit\_text\_rounded" está sendo incluido o arquivo do diretório  
 "drawable" que configura esta formatação arredondada para a aplicação.  
 -->  
  
 <EditText  
 android:id="@+id/editText\_Email"  
 android:layout\_width="0dp"  
 android:layout\_height="50dp"  
 android:layout\_marginTop="8dp"  
 android:background="@drawable/bg\_edit\_text\_rounded"  
 android:ems="10"  
 android:hint="Email"  
 android:inputType="textEmailAddress"  
 android:paddingLeft="10dp"  
 app:layout\_constraintEnd\_toEndOf="@+id/editText\_Nome"  
 app:layout\_constraintStart\_toStartOf="@+id/editText\_Nome"  
 app:layout\_constraintTop\_toBottomOf="@+id/editText\_Nome" />  
  
 <EditText  
 android:id="@+id/editText\_Senha"  
 android:layout\_width="0dp"  
 android:layout\_height="50dp"  
 android:layout\_marginTop="8dp"  
 android:background="@drawable/bg\_edit\_text\_rounded"  
 android:paddingLeft="10dp"  
 android:hint="Senha"  
 android:ems="10"  
 android:inputType="textPassword"  
 android:textSize="18sp"  
 app:layout\_constraintEnd\_toEndOf="@+id/editText\_Email"  
 app:layout\_constraintStart\_toStartOf="@+id/editText\_Email"  
 app:layout\_constraintTop\_toBottomOf="@+id/editText\_Email" />  
  
 <Button  
 android:id="@+id/btn\_Cadastrar"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginStart="8dp"  
 android:layout\_marginLeft="8dp"  
 android:layout\_marginTop="16dp"  
 android:layout\_marginEnd="8dp"  
 android:layout\_marginRight="8dp"  
 android:background="@drawable/bg\_edit\_text\_rounded"  
 android:backgroundTint="@android:color/holo\_green\_dark"  
 android:text="@string/cadastrar"  
 android:textColor="@android:color/white"  
 android:textSize="24sp"  
 android:textStyle="bold"  
 app:layout\_constraintEnd\_toEndOf="@+id/editText\_Senha"  
 app:layout\_constraintStart\_toStartOf="@+id/editText\_Senha"  
 app:layout\_constraintTop\_toBottomOf="@+id/editText\_Senha" />  
  
 <Button  
 android:id="@+id/idBtnFotoUsuario"  
 android:layout\_width="130dp"  
 android:layout\_height="130dp"  
 android:layout\_marginTop="8dp"  
 android:text="@string/img\_Foto"  
 android:textColor="@android:color/white"  
 android:textStyle="bold"  
 android:background="@drawable/bg\_img\_rounded"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <de.hdodenhof.circleimageview.CircleImageView  
 android:id="@+id/idImg\_Foto"  
 android:layout\_width="130dp"  
 android:layout\_height="130dp"  
 app:civ\_border\_width="2dp"  
 app:civ\_border\_color="@android:color/holo\_green\_dark"  
 app:layout\_constraintBottom\_toBottomOf="@+id/idBtnFotoUsuario"  
 app:layout\_constraintEnd\_toEndOf="@+id/idBtnFotoUsuario"  
 app:layout\_constraintStart\_toStartOf="@+id/idBtnFotoUsuario"  
 app:layout\_constraintTop\_toTopOf="@+id/idBtnFotoUsuario"  
 app:layout\_constraintVertical\_bias="0.0" />  
  
  
</androidx.constraintlayout.widget.ConstraintLayout>

Arquivo layout “**item\_user.xml**” da aplicação do diretório “**layout**”

<?xml version="1.0" encoding="utf-8"?>  
  
<!-- Importante observar os logo abaixo:  
 "android:layout\_width="match\_parent"  
 "android:layout\_height="wrap\_content"  
 -->  
<androidx.constraintlayout.widget.ConstraintLayout  
 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:orientation="vertical"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
  
 <ImageView  
 android:id="@+id/idImageAvatar"  
 android:layout\_width="60dp"  
 android:layout\_height="50dp"  
 android:layout\_marginStart="4dp"  
 android:layout\_marginLeft="4dp"  
 android:layout\_marginTop="8dp"  
 android:layout\_marginBottom="8dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.0"  
 tools:srcCompat="@tools:sample/avatars[4]" />  
  
 <TextView  
 android:id="@+id/textView"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginStart="4dp"  
 android:layout\_marginLeft="4dp"  
 android:layout\_marginEnd="8dp"  
 android:layout\_marginRight="8dp"  
 android:text=""  
 android:textSize="16dp"  
 android:textStyle="bold"  
 app:layout\_constraintBottom\_toBottomOf="@+id/idImageAvatar"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.0"  
 app:layout\_constraintStart\_toEndOf="@+id/idImageAvatar"  
 app:layout\_constraintTop\_toTopOf="@+id/idImageAvatar"  
 app:layout\_constraintVertical\_bias="0.0" />  
  
</androidx.constraintlayout.widget.ConstraintLayout>

Arquivo layout “**menu.xml**” da aplicação do diretório “**menu**”

<?xml version="1.0" encoding="utf-8"?>  
<menu xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto">  
  
 <!--## Aqui teremos a criação de menus da página ou layout "MessagesActivity" ##-->  
 <item  
 android:id="@+id/idContatos"  
 android:title="@string/contatos"  
 app:showAsAction="collapseActionView">  
 </item>  
  
 <item  
 android:id="@+id/idSair"  
 android:title="@string/sair"  
 app:showAsAction="collapseActionView">  
 </item>  
  
</menu>

Arquivo layout “**colors.xml**” da aplicação do diretório “**values**”

<?xml version="1.0" encoding="utf-8"?>  
<resources>  
 <color name="colorPrimary">#008577</color>  
 <color name="colorPrimaryDark">#00574B</color>  
 <color name="colorAccent">#D81B60</color>  
 <!--  
 Aqui abaixo está sendo criada uma cor nova para este arquivo e para a aplicação  
 -->  
 <color name="bgColor">#DCDCDC</color>  
</resources>

Arquivo layout “**strings.xml**” da aplicação do diretório “**values**”

<resources>  
 <string name="app\_name">ChatFirebase</string>  
 <string name="signIn">Não tem uma Conta? Crie Aqui!</string>  
 <string name="foto">Img\_Foto</string>  
 <string name="Foto">Img\_Foto</string>  
 <string name="img\_Foto">Foto</string>  
 <string name="enterinter">Login</string>  
 <string name="enter" translatable="false">Login</string>  
 <string name="cadastrar">Cadastrar</string>  
 <string name="contatos">Contatos</string>  
 <string name="sair">Sair</string>  
</resources>

Arquivo layout “**styles.xml**” da aplicação do diretório “**values**”

<resources>  
 <string name="app\_name">Recyclerview Dominio</string>  
</resources>

Arquivo layout “**styles.xml**” da aplicação do diretório “**values**”

<resources>  
  
 <!-- Base application theme. -->  
 <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">  
 <!-- Customize your theme here. -->  
 <item name="colorPrimary">@color/colorPrimary</item>  
 <item name="colorPrimaryDark">@color/colorPrimaryDark</item>  
 <item name="colorAccent">@color/colorAccent</item>  
 </style>  
  
</resources>

Arquivo construtor “**build.gradle” (Project: Nome do Projeto)** da aplicação do diretório “**Gradle Scripts**”

// Top-level build file where you can add configuration options common to all sub-projects/modules.  
  
buildscript {  
 repositories {  
 google()  
 jcenter()  
   
 }  
 dependencies {  
 classpath 'com.android.tools.build:gradle:3.5.3'  
   
 // NOTE: Do not place your application dependencies here; they belong  
 // in the individual module build.gradle files  
  
 //Aqui abaixo entra o código de dependência do Firebase da aplicação no diretório "build.gradle(Project: Nome do App)"  
 classpath 'com.google.gms:google-services:4.3.2'  
 }  
}  
  
allprojects {  
 repositories {  
 google()  
 jcenter()  
   
 }  
}  
  
task clean(type: Delete) {  
 delete rootProject.buildDir  
}

Arquivo construtor “**build.gradle” (Module: app)** da aplicação do diretório “**Gradle Scripts**”

apply plugin: 'com.android.application'  
  
android {  
 compileSdkVersion 29  
 buildToolsVersion "29.0.2"  
 defaultConfig {  
 applicationId "com.ramerips\_cursos"  
 minSdkVersion 15  
 targetSdkVersion 29  
 versionCode 1  
 versionName "1.0"  
 testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  
 }  
 buildTypes {  
 release {  
 minifyEnabled false  
 proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  
 }  
 }  
}  
  
dependencies {  
 implementation fileTree(dir: 'libs', include: ['\*.jar'])  
 implementation 'androidx.appcompat:appcompat:1.0.2'  
 implementation 'androidx.constraintlayout:constraintlayout:1.1.3'  
  
 //Aqui abaixo entra o código de "analytics" do Firebase da aplicação no diretório "build.gradle(Module: app)"  
 //Aqui abaixo tive que mudar de "17.2.0 para 16.0.0" pois sem a mudança não funcionava  
 implementation 'com.google.firebase:firebase-analytics:16.0.0'  
 //Aqui abaixo entra o código de dependência de autenticação do "Firebase Authentication"  
 //Aqui abaixo tive que mudar de "19.1.0 para 15.0.0" pois sem a mudança não funcionava  
 implementation 'com.google.firebase:firebase-auth:15.0.0'  
 //Aqui abaixo temos a dependência da "circleimageview", ou seja, imagens em círculos  
 //https://github.com/hdodenhof/CircleImageView -> Site desta dependência no GitHub  
 implementation 'de.hdodenhof:circleimageview:3.0.2'  
 //Aqui abaixo temos a dependência da "firebase-storage", ou seja, dependência de armazenamentos de imagens e vídeos  
 //https://firebase.google.com/docs/storage/android/start -> Site desta dependência no Firebase  
 implementation 'com.google.firebase:firebase-storage:16.0.1'  
 //Aqui abaixo temos a dependência da "firebase-firestore", ou seja, dependência de armazenamentos de dados ou arquivos  
 //https://firebase.google.com/docs/android/setup?hl=pt-br#available-libraries -> Site desta dependência no Firebase  
 implementation 'com.google.firebase:firebase-firestore:17.0.1'  
 //Aqui abaixo temos as dependências do sistema de lista de visualização "recyclerview:recyclerview"  
 implementation 'androidx.recyclerview:recyclerview:1.1.0'  
  
 testImplementation 'junit:junit:4.12'  
 androidTestImplementation 'androidx.test.ext:junit:1.1.0'  
 androidTestImplementation 'androidx.test.espresso:espresso-core:3.1.1'  
  
  
}  
  
//Aqui abaixo entra o código de "plugin" do Firebase da aplicação no diretório "build.gradle(Module: app)"  
apply plugin: 'com.google.gms.google-services'