PRA3 – Prof. Adriano Domingues

Tutorial CameraX – Android Java

A biblioteca **CameraX** é a mais nova biblioteca do Jetpack para desenvolvimento da funcionalidade de capturar imagens, agora com possibilidade de realizar uma pré-visualização da imagem e também recursos de análise e tratamento da imagem.

Recursos do CameraX:

- Visualização: Apresenta uma imagem na tela.
- Análise da Imagem: Utilizar a imagem em serviços do MLKit (Reconhecimento de objetos, aprendizagem de máquina, etc..).
- Captura de Imagem: Tirar fotos com alta resolução.

Neste tutorial vamos abordar e entender como funciona o Preview e o ImageCapture.

Siga o tutorial, altere os arquivos indicados adicionando os códigos demonstrados.

Android Manifest

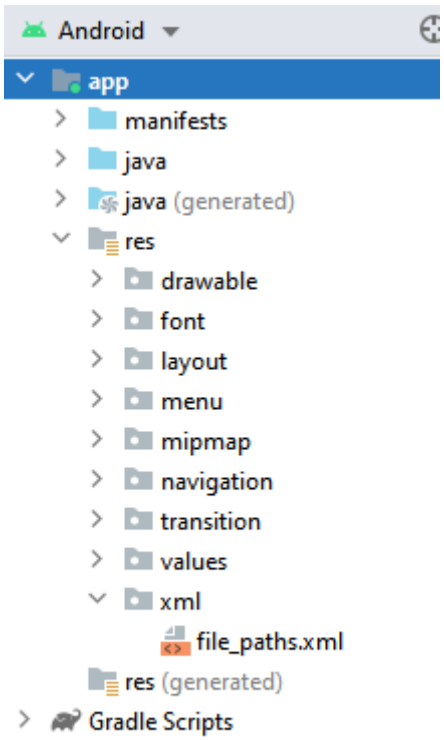
Adicione as permissões abaixo no arquivo manifesto, abaixo vamos requerer do sistema operacional a utilização da câmera, do armazenamento e da internet:

```
<uses-feature android:name="android.hardware.camera.any" />
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission
android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.INTERNET"/>
```

Também no arquivo de manifesto precisamos informar o Provedor de Conteúdo, que irá direcionar o caminho de gravação das imagens no aparelho.

```
<provider
    android:name="androidx.core.content.FileProvider"
    android:authorities="${applicationId}.provider"
    android:exported="false"
    android:grantUriPermissions="true">
    <meta-data
        android:name="android.support.FILE_PROVIDER_PATHS"
        android:resource="@xml/file_paths" />
</provider>
```

Em seguida crie uma pasta chamada ***xml*** dentro da pasta ***res*** do projeto e nesta pasta crie um arquivo **XML** chamado ***file_paths.xml***.



Adicione o código abaixo no arquivo file_paths.xml criado na pasta res/xml:

```
<?xml version="1.0" encoding="utf-8"?>
<paths xmlns:android="http://schemas.android.com/apk/res/android">
    <external-path
        name="external"
        path="." />
    <external-files-path
        name="external_files"
        path="." />
    <cache-path
        name="cache"
        path="." />
    <external-cache-path
        name="external_cache"
        path="." />
    <files-path
        name="files"
        path="." />
</paths>
```

Segue abaixo o layout da tela XML que será utilizada neste projeto:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.appcompat.widget.LinearLayoutCompat
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@color/grey_dark"
    style="@style/Theme.GEOPLENO_MOVEL"
    tools:context=".ui.pessoa.CameraFragment">

    <androidx.cardview.widget.CardView
        android:elevation="8dp"
        app:cardElevation="8dp"
        app:cardMaxElevation="32dp"
        app:cardCornerRadius="32dp"
        android:layout_margin="8dp"
        style="@style/Theme.GEOPLENO_MOVEL"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <androidx.appcompat.widget.LinearLayoutCompat
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="top|center"
            android:orientation="vertical">

        <androidx.cardview.widget.CardView
            android:elevation="8dp"
            app:cardElevation="8dp"
            app:cardMaxElevation="32dp"
            app:cardCornerRadius="32dp"
            android:layout_margin="8dp"
            android:layout_gravity="top|center"
            style="@style/Theme.GEOPLENO_MOVEL"
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <androidx.appcompat.widget.LinearLayoutCompat
                android:orientation="vertical"
                android:layout_width="match_parent"
                android:layout_height="match_parent">

            <androidx.coordinatorlayout.widget.CoordinatorLayout
                android:layout_gravity="center"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content">

                <androidx.camera.view.PreviewView
                    android:id="@+id/viewFinder"
                    android:visibility="visible"
                    android:layout_gravity="top|center"
                    android:layout_margin="32dp"
                    android:layout_width="400dp"
                    android:layout_height="700dp"/>
```

Continuação do layout iniciado acima:

```
<androidx.appcompat.widget.AppCompatImageView
                            android:id="@+id/ivFoto"
                            android:visibility="invisible"
                            android:layout_gravity="top|center"
                            android:layout_margin="32dp"
                            android:layout_width="400dp"
                            android:layout_height="700dp"/>


<com.google.android.material.floatingactionbutton.FloatingActionButt
on
                            android:id="@+id/btn_galeria"
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"
                            android:layout_gravity="bottom|center"
                            android:layout_marginRight="148dp"
                            android:layout_marginBottom="16dp"
                            android:adjustViewBounds="true"
                            android:backgroundTint="@color/white"

android:src="@drawable/ic_action_galeria"
                            app:elevation="8dp"
                            app:fabSize="auto" />


<com.google.android.material.floatingactionbutton.FloatingActionButt
on
                            android:id="@+id/btn_confirma_foto"
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"
                            android:layout_gravity="bottom|center"
                            android:layout_marginRight="148dp"
                            android:layout_marginBottom="16dp"
                            android:adjustViewBounds="true"
                            android:backgroundTint="@color/white"

android:src="@drawable/ic_action_confirmar"
                            android:visibility="invisible"
                            app:elevation="8dp"
                            app:fabSize="auto" />


<com.google.android.material.floatingactionbutton.FloatingActionButt
on
                            android:id="@+id/btn_troca_camera"
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"
                            android:layout_gravity="top|end"
                            android:layout_marginRight="16dp"
                            android:layout_marginTop="42dp"
                            android:adjustViewBounds="true"
                            android:backgroundTint="@color/white"
                            android:src="@drawable/ic_action_troca"
                            android:visibility="visible"
                            app:elevation="8dp"
                            app:fabSize="auto" />
```

Parte final do layout do fragmento CameraFragment:

```
<com.google.android.material.floatingactionbutton.FloatingActionBut
ton
                            android:id="@+id/btn_cancela_foto"
                            app:elevation="8dp"
android:src="@drawable/ic_action_cancelar"
                            android:adjustViewBounds="true"
                            app:fabSize="auto"
                            android:layout_marginLeft="148dp"
                            android:layout_marginBottom="16dp"
                            android:layout_gravity="bottom|center"
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"/>
<com.google.android.material.floatingactionbutton.FloatingActionBut
ton
                            android:id="@+id/btn_tirar_foto"
                            app:elevation="8dp"
android:src="@drawable/ic_action_tirar_foto"
                            android:adjustViewBounds="true"
                            app:fabSize="auto"
                            android:layout_marginLeft="148dp"
                            android:layout_marginBottom="16dp"
                            android:layout_gravity="bottom|center"
                            android:backgroundTint="@color/white"
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"/>

                        <androidx.appcompat.widget.AppCompatButton
                            android:id="@+id/btn_continua_pessoa2"
style="@style/Widget.AppCompat.Button.Borderless.Colored"
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"
                            android:layout_gravity="bottom|center"
                            android:layout_marginLeft="32dp"
                            android:layout_marginTop="8dp"
                            android:layout_marginRight="32dp"
                            android:layout_marginBottom="96dp"
                            android:background="@color/grey_dark"
                            android:visibility="invisible"
                            android:padding="8dp"
                            android:text="PRÓXIMA"
                            android:textColor="@color/white" />

</androidx.coordinatorlayout.widget.CoordinatorLayout>

                    <androidx.appcompat.widget.LinearLayoutCompat
                        android:orientation="horizontal"
                        android:padding="8dp"
                        android:layout_gravity="center"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content">
                    </androidx.appcompat.widget.LinearLayoutCompat>
                </androidx.appcompat.widget.LinearLayoutCompat>
            </androidx.cardview.widget.CardView>
        </androidx.appcompat.widget.LinearLayoutCompat>
    </androidx.cardview.widget.CardView>
</androidx.appcompat.widget.LinearLayoutCompat>
```

Adicione as dependências abaixo para utilização da biblioteca CameraX no arquivo build.gradle (app) do seu projeto:

```
def camerax_version = '1.0.0-rc01'
implementation "androidx.camera:camera-camera2:$camerax_version"
implementation "androidx.camera:camera-lifecycle:$camerax_version"
implementation "androidx.camera:camera-view:1.0.0-alpha20"
```

Altere a classe que será responsável por manipular a câmera com as declarações abaixo:

```
private ImageCapture imageCapture;
File outputDirectory;
ExecutorService cameraExecutor;
private static String TAG = "CameraXBasic";
private static String FILENAME_FORMAT = "yyyy-MM-dd-HH-mm-ss-SSS";
private static int REQUEST_CODE_PERMISSIONS = 10;
private static String REQUIRED_PERMISSIONS =
Manifest.permission.CAMERA;
//Declaração de constantes
private static final int REQUEST_TAKE_PHOTO = 1;
private static final int PERMISSAO_REQUEST = 2;
private static final int PEGA_FOTO = 3;
//Declaração de objetos criados nas telas activity e content
//private AppCompatImageView imageView;
//private AppCompatTextView textView;
private String currentPhotoPath;
private Bitmap bitmap;
private Uri photoURI;
File photoFile;
Camera camera;
CameraSelector cameraSelector;
Preview preview;
ProcessCameraProvider cameraProvider;
PreviewView previewView;
AppCompatImageView ivFoto;
FloatingActionButton botaoGaleria, botaoTirarFoto,
        botaoConfirmar, botaoCancelar, botaoTrocarCamera;
AppCompatButton botaoProximo;
private int tipoCamera = 2;
```

Este exemplo foi aplicado em um Fragmento, então se você deseja inserir em uma Atividade deve realizar algumas adaptações, ou seguir o tutorial oficial do developer.android.com que já esta com esta abordagem:

```
public static CameraFragment newInstance() {
    return new CameraFragment();
}

@Override
public void onCreate(@Nullable Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    }
```

Os métodos abaixo estão preparados para capturar a imagem através da câmera ou da galeria.

O método startCamera() constrói os objetos de pré-visualização, de captura, de seleção de câmera e do provedor de conteúdo.

```
private void startCamera(View v, int tipoCamera) {

        ListenableFuture cameraProviderFuture =
                ProcessCameraProvider.getInstance(v.getContext());

        cameraProviderFuture.addListener(() -> {
            try {
                // Camera provider is now guaranteed to be available
                cameraProvider = (ProcessCameraProvider)
cameraProviderFuture.get();

                // Set up the view finder use case to display camera
preview
                preview = new Preview.Builder().build();

                // Set up the capture use case to allow users to take
photos
                imageCapture = new ImageCapture.Builder()

.setCaptureMode(ImageCapture.CAPTURE_MODE_MAXIMIZE_QUALITY)
                        .build();

                if(tipoCamera == 1) {
                    // Choose the camera by requiring a lens facing
                    cameraSelector = new CameraSelector.Builder()

.requireLensFacing(CameraSelector.LENS_FACING_FRONT)
                            .build();
                } else if(tipoCamera == 2){
                    // Choose the camera by requiring a lens facing
                    cameraSelector = new CameraSelector.Builder()

.requireLensFacing(CameraSelector.LENS_FACING_BACK)
                            .build();
                }

                // Attach use cases to the camera with the same
lifecycle owner
                camera = cameraProvider.bindToLifecycle(
                        ((LifecycleOwner) this),
                        cameraSelector,
                        preview,
                        imageCapture);

                // Connect the preview use case to the previewView

preview.setSurfaceProvider(previewView.getSurfaceProvider());
            } catch (InterruptedException | ExecutionException e) {
                // Currently no exceptions thrown.
cameraProviderFuture.get()
                // shouldn't block since the listener is being called,
so no need to
                // handle InterruptedException.
            }
        }, ContextCompat.getMainExecutor(v.getContext()));
    }
```

O método abaixo será utilizado para solicitar permissão de utilização da câmera:

```
private boolean allPermissionsGranted(View v) {
    if (ContextCompat.checkSelfPermission(
            v.getContext(), Manifest.permission.CAMERA) ==
PackageManager.PERMISSION_GRANTED) {
        return true;
    }
    return false;
}
```

O método onCreateView() executa a construção dos objetos de vínculo com a tela (XML), e chamadas de métodos de construção de objetos do CameraX.

```
@Override
public View onCreateView(@NonNull LayoutInflater inflater,
@Nullable ViewGroup container,
                         @Nullable Bundle savedInstanceState) {
    View view = inflater.inflate(R.layout.fragment_pessoa2,
container, false);

    previewView = view.findViewById(R.id.viewFinder);
    ivFoto = view.findViewById(R.id.ivFoto);
    botaoGaleria = view.findViewById(R.id.btn_galeria);
    botaoTirarFoto = view.findViewById(R.id.btn_tirar_foto);
    botaoConfirmar = view.findViewById(R.id.btn_confirma_foto);
    botaoCancelar = view.findViewById(R.id.btn_cancela_foto);
    botaoProximo = view.findViewById(R.id.btn_continua_pessoa2);
    botaoTrocarCamera = view.findViewById(R.id.btn_troca_camera);

    // Request camera permissions
    if (allPermissionsGranted(view)) {

    } else {
        permissoesAcesso();
    }

    startCamera(view, tipoCamera);

    botaoProximo.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {

        }
    });
    return view;
}
```

O método onActivityCreated() é o responsável por construir os botões de acionamento:

```
@Override
public void onActivityCreated(@Nullable Bundle savedInstanceState) {
    super.onActivityCreated(savedInstanceState);
    mViewModel = new
ViewModelProvider(this).get(Pessoa2ViewModel.class);
    // TODO: Use the ViewModel

    // Preparar botões de Galeria e Câmera
    //Botão para editar foto do produto
    botaoGaleria.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Snackbar.make(v, "Acessando Galeria!",
Snackbar.LENGTH_LONG)
                    .setAction("Action", null).show();
            // Acesso à Galeria
            acessarGaleria();
            if(ivFoto != null){
                ivFoto.setVisibility(View.VISIBLE);
                previewView.setVisibility(View.INVISIBLE);
                botaoGaleria.setVisibility(View.INVISIBLE);
                botaoTirarFoto.setVisibility(View.INVISIBLE);
                botaoConfirmar.setVisibility(View.VISIBLE);
                botaoCancelar.setVisibility(View.VISIBLE);
            }

        }
    });

//Botão para acionamento da câmera
botaoTirarFoto.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Snackbar.make(v, "Capturando Imagem!", Snackbar.LENGTH_LONG)
                .setAction("Action", null).show();
        //Chamada do método tirarFoto()
        try {
            tirarFoto2(v);
        } catch (IOException e) {
            e.printStackTrace();
        }
        if(ivFoto != null){
            ivFoto.setVisibility(View.VISIBLE);
            previewView.setVisibility(View.INVISIBLE);
            botaoGaleria.setVisibility(View.INVISIBLE);
            botaoTirarFoto.setVisibility(View.INVISIBLE);
            botaoConfirmar.setVisibility(View.VISIBLE);
            botaoCancelar.setVisibility(View.VISIBLE);
        }
    }
});
```

Ainda no método onActivityCreated() vamos implementar os botões de troca de câmera (frontal ou traseira), cancelamento ou confirmação da imagem capturada:

```
botaoTrocarCamera.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            cameraProvider.unbindAll();

            if(tipoCamera == 1) {
                tipoCamera = 2;
                // Choose the camera by requiring a lens facing
                startCamera(v, tipoCamera);
            } else if(tipoCamera == 2){
                tipoCamera = 1;
                // Choose the camera by requiring a lens facing
                startCamera(v, tipoCamera);
            }

        }
    });
}

botaoCancelar.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        ivFoto.setVisibility(View.INVISIBLE);
        previewView.setVisibility(View.VISIBLE);
        botaoGaleria.setVisibility(View.VISIBLE);
        botaoTirarFoto.setVisibility(View.VISIBLE);
        botaoConfirmar.setVisibility(View.INVISIBLE);
        botaoCancelar.setVisibility(View.INVISIBLE);
        botaoProximo.setVisibility(View.INVISIBLE);
    }
});

botaoConfirmar.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        botaoProximo.setFocusable(true);
        botaoProximo.setVisibility(View.VISIBLE);
    }
});
```

O método criado abaixo utiliza o objeto do tipo ImageCapture para obter uma imagem de alta resolução e salvando a mesma na galeria do dispositivo:

```
private void tirarFoto2(View v) throws IOException {
        // Get a stable reference of the modifiable image capture
use case
        //imageCapture = imageCapture ?: return

        // Create time-stamped output file to hold the image
        File photoFile = new File(
                outputDirectory,
String.valueOf(createImageFile()));

        // Create output options object which contains file +
metadata
        ImageCapture.OutputFileOptions outputOptions =
                new
ImageCapture.OutputFileOptions.Builder(photoFile).build();

        // Set up image capture listener, which is triggered after
photo has
        // been taken

        imageCapture.takePicture(outputOptions,
ContextCompat.getMainExecutor(v.getContext()),
                new ImageCapture.OnImageSavedCallback() {
                    @Override
                    public void
onImageSaved(ImageCapture.OutputFileResults outputFileResults) {
                        // insert your code here.
                        Uri savedUri = Uri.fromFile(photoFile);
//                        Snackbar.make(v, "Foto capturada " +
//                                savedUri.getPath() + "!",
Snackbar.LENGTH_LONG).show();
                        // Chamada do método de adição da imagem na
galeria
                        galleryAddPic(savedUri);
                        // Definição das dimensões da imagem
                        setPic();
                    }

                    @Override
                    public void onError(ImageCaptureException
error) {
                        // insert your code here.
                        Snackbar.make(v, "Erro ao capturar foto " +
                                error.getMessage() + "!",
Snackbar.LENGTH_LONG).show();
                    }
                }
        );
    }
```

O método abaixo inicia o acesso à galeria do dispositivo:

```
//Método para acessar a galeria
private void acessarGaleria() {
    Intent intentPegaFoto = new Intent(Intent.ACTION_PICK,
            MediaStore.Images.Media.INTERNAL_CONTENT_URI);
    startActivityForResult(intentPegaFoto, PEGA_FOTO);
}
```

O método permissõesAcesso() realiza o processo de solicitação de permissões já declaradas no arquivo de manifesto:

```
private void permissoesAcesso() {
    //Condicional para controle de permissões
    // Verifica se há permissão para leitura de arquivos
    if (ContextCompat.checkSelfPermission(this.getContext(),
            android.Manifest.permission.READ_EXTERNAL_STORAGE) !=
            PackageManager.PERMISSION_GRANTED) {
        if
(ActivityCompat.shouldShowRequestPermissionRationale(getActivity(),
                android.Manifest.permission.READ_EXTERNAL_STORAGE))
{
        } else {
            ActivityCompat.requestPermissions(getActivity(),
                    new
String[]{android.Manifest.permission.READ_EXTERNAL_STORAGE},
PERMISSAO_REQUEST);
        }
    }
    //Verifica se há permissões para escrita de arquivos
    if (ContextCompat.checkSelfPermission(this.getContext(),
            android.Manifest.permission.WRITE_EXTERNAL_STORAGE) !=
PackageManager.PERMISSION_GRANTED) {
        if
(ActivityCompat.shouldShowRequestPermissionRationale(getActivity(),
                android.Manifest.permission.WRITE_EXTERNAL_STORAGE))
{
        } else {
            ActivityCompat.requestPermissions(getActivity(),
                    new
String[]{android.Manifest.permission.WRITE_EXTERNAL_STORAGE},
PERMISSAO_REQUEST);
        }
    }
    //Verifica se há permissões para escrita de arquivos
    if (ContextCompat.checkSelfPermission(this.getContext(),
            Manifest.permission.CAMERA) !=
PackageManager.PERMISSION_GRANTED) {
        if
(ActivityCompat.shouldShowRequestPermissionRationale(getActivity(),
                android.Manifest.permission.CAMERA)) {
        } else {
            ActivityCompat.requestPermissions(getActivity(),
                    new
String[]{android.Manifest.permission.CAMERA}, PERMISSAO_REQUEST);
        }
    }
}
```

O método createImageFile() define o formato e no nome do arquivo que será criado para receber a imagem capturada.

```
private File createImageFile() throws IOException {
    // Create an image file name
    String timeStamp = new
SimpleDateFormat("yyyyMMdd_HHmmss").format(new Date());
    String imageFileName = "JPEG_" + timeStamp + "_";
    File storageDir =
this.getContext().getExternalFilesDir(Environment.DIRECTORY_PICTURES
);
    File image = File.createTempFile(
            imageFileName,  /* prefix */
            ".jpg",         /* suffix */
            storageDir      /* directory */
    );

    // Save a file: path for use with ACTION_VIEW intents
    currentPhotoPath = image.getAbsolutePath();
    return image;
}
```

Este método inclui a imagem capturada na galeria do dispositivo.

```
//Método para inclusão de imagem na galeria
private void galleryAddPic(Uri photoURI) {
    //File f = new File(currentPhotoPath);
    this.getActivity().sendBroadcast(
            new Intent(Intent.ACTION_MEDIA_SCANNER_SCAN_FILE,
                    photoURI));
}
```

Este método garante a solicitação da permissão de gravação no dispositivo:

```
// TODO -
public boolean isExternalStorageWritable() {
    String state = Environment.getExternalStorageState();
    if (Environment.MEDIA_MOUNTED.equals(state)) {
        return true;
    }
    return false;
}
```

O método setPic() realiza o tratamento da imagem e da rotação da imagem:

```
//Método para definição das dimensões da imagem
private void setPic() {
    // Obtendo as dimensões da imagem para a View
    int targetW = ivFoto.getWidth();
    int targetH = ivFoto.getHeight();

    // Obter as dimensões do Bitmap
    BitmapFactory.Options bmOptions = new BitmapFactory.Options();
    //bmOptions.inJustDecodeBounds = true;
    //BitmapFactory.decodeFile(currentPhotoPath, bmOptions);
    int photoW = bmOptions.outWidth;
    int photoH = bmOptions.outHeight;
    // Determina como diminuir a escala da imagem
    int scaleFactor = Math.min(photoW / targetW, photoH / targetH);

    // Decodifica o arquivo de imagem para o Bitmap que preencherá
a View
    bmOptions.inJustDecodeBounds = false;
    bmOptions.inSampleSize = scaleFactor;
    bmOptions.inPurgeable = true;

    // Cria o bitmap da imagem capturada
    bitmap = BitmapFactory.decodeFile(currentPhotoPath, bmOptions);

    // Declara objeto vetor e rotaciona em 90 graus a imagem
    Matrix matrix = new Matrix();
    matrix.postRotate(270);
    if(tipoCamera == 2){
        matrix.invert(matrix);
    }
    //matrix.preRotate(45);
    Bitmap rotated = Bitmap.createBitmap(bitmap, 0, 0,
bitmap.getWidth(),
            bitmap.getHeight(), matrix, true);
    // Devolve imagem para o ImageView
    ivFoto.setImageBitmap(rotated);

    // Apresenta a imagem na tela
    //ivFoto.setImageBitmap(bitmap);
}
```

Esta método obtêm o caminho da pasta onde as imagem são salvas.

```
// externo
public File getAlbumStorageDir(String albumName) {
    // Get the directory for the user's public pictures directory.
    File file = new
File(Environment.getExternalStoragePublicDirectory(
            Environment.DIRECTORY_DCIM), albumName);
    if (!file.mkdirs()) {
        Log.e(TAG, "Directory not created");
    }
    return file;
}
```

Sobrecarga do método escrito acima, mas com outro caminho de armazenamento:

```
// local
public File getAlbumStorageDir(Context context, String albumName) {
    // Get the directory for the app's private pictures directory.
    File file = new File(context.getExternalFilesDir(
            Environment.DIRECTORY_DCIM), albumName);
    if (!file.mkdirs()) {
        Log.e(TAG, "Directory not created");
    }
    return file;
}
```

Método que retorna o resultado da chamada da câmera pela Intent :

```
@Override
public void onActivityResult(int requestCode, int resultCode,
@Nullable Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    //imageView =
this.getActivity().findViewById(R.id.foto_detalhe_produto);
    if (requestCode == REQUEST_TAKE_PHOTO && resultCode ==
RESULT_OK) {
        // Chamada do método de adição da imagem na galeria
        galleryAddPic(photoURI);
        // Definição das dimensões da imagem
        setPic();
    } else if (requestCode == PEGA_FOTO && resultCode == RESULT_OK)
{
        //Captura caminho da imagem selecionada
        Uri imagemSelecionada = data.getData();

        // declara um stream (seguimento de dados) para ler a
imagem recuperada do SD Card
        InputStream inputStream = null;
        // recuperando a sequencia de entrada, baseada no caminho
(uri) da imagem
        try {
            inputStream = this.getActivity().getContentResolver()
                    .openInputStream(imagemSelecionada);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        // recuperando um bitmap do stream
        bitmap = BitmapFactory.decodeStream(inputStream);
        // Vínculo do objeto ImageView
        //imageView = (AppCompatImageView) this.getActivity()
        //        .findViewById(R.id.foto_detalhe_produto);
        // Reduz imagem e configura apresentação
        Bitmap bitmapReduzido = Bitmap
                .createScaledBitmap(bitmap, 1080, 1080, true);
        ivFoto.setImageBitmap(bitmapReduzido);
        ivFoto.setScaleType(AppCompatImageView.ScaleType.FIT_XY);
    }
}
```

Abaixo temos a classe de captura e gravação de imagem completa:

```
import androidx.appcompat.widget.AppCompatButton;
import androidx.appcompat.widget.AppCompatImageView;
import androidx.camera.core.Camera;
import androidx.camera.core.CameraSelector;
import androidx.camera.core.ImageCapture;
import androidx.camera.core.ImageCaptureException;
import androidx.camera.core.Preview;
import androidx.camera.lifecycle.ProcessCameraProvider;
import androidx.camera.view.PreviewView;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import androidx.core.content.FileProvider;
import androidx.lifecycle.LifecycleOwner;
import androidx.lifecycle.ViewModelProvider;

import android.Manifest;
import android.content.ContentValues;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Matrix;
import android.net.Uri;
import android.os.Bundle;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;

import android.os.Environment;
import android.provider.MediaStore;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.OrientationEventListener;
import android.view.Surface;
import android.view.View;
import android.view.ViewGroup;

import
com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.android.material.snackbar.Snackbar;
import com.google.common.util.concurrent.ListenableFuture;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;
import java.util.Objects;
import java.util.concurrent.ExecutionException;
import java.util.concurrent.ExecutorService;

import br.com.alladin.geopleno_movel.R;

import static android.app.Activity.RESULT_OK;
import static android.content.ContentValues.TAG;
```

```
public class Pessoa2Fragment extends Fragment {

    private Pessoa2ViewModel mViewModel;
    private String nome;
    private String email;
    private String telefone;
    private int tipoPessoa;
    private ImageCapture imageCapture;
    File outputDirectory;
    ExecutorService cameraExecutor;

    private static String TAG = "CameraXBasic";
    private static String FILENAME_FORMAT = "yyyy-MM-dd-HH-mm-ss-SSS";
    private static int REQUEST_CODE_PERMISSIONS = 10;
    private static String REQUIRED_PERMISSIONS =
Manifest.permission.CAMERA;
    //Declaração de constantes
    private static final int REQUEST_TAKE_PHOTO = 1;
    private static final int PERMISSAO_REQUEST = 2;
    private static final int PEGA_FOTO = 3;
    //Declaração de objetos criados nas telas activity e content
    //private AppCompatImageView imageView;
    //private AppCompatTextView textView;
    private String currentPhotoPath;
    private Bitmap bitmap;
    private Uri photoURI;
    File photoFile;
    Camera camera;
    CameraSelector cameraSelector;
    Preview preview;
    ProcessCameraProvider cameraProvider;
    PreviewView previewView;
    AppCompatImageView ivFoto;
    FloatingActionButton botaoGaleria, botaoTirarFoto,
            botaoConfirmar, botaoCancelar, botaoTrocarCamera;
    AppCompatButton botaoProximo;
    private int tipoCamera = 2;

    public static Pessoa2Fragment newInstance() {
        return new Pessoa2Fragment();
    }

    @Override
    public void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        if (getArguments() != null) {
            nome = getArguments().getString("nome");
            email = getArguments().getString("email");
            telefone = getArguments().getString("telefone");
            tipoPessoa = getArguments().getInt("tipoPessoa", -1);
        }
    }

    private void startCamera(View v, int tipoCamera) {

        ListenableFuture cameraProviderFuture =
                ProcessCameraProvider.getInstance(v.getContext());

        cameraProviderFuture.addListener(() -> {
```

```
            try {
                // Camera provider is now guaranteed to be available
                cameraProvider = (ProcessCameraProvider)
cameraProviderFuture.get();

                // Set up the view finder use case to display camera
preview
                preview = new Preview.Builder().build();

                // Set up the capture use case to allow users to take
photos
                imageCapture = new ImageCapture.Builder()

.setCaptureMode(ImageCapture.CAPTURE_MODE_MAXIMIZE_QUALITY)
                        .build();

//                OrientationEventListener orientationEventListener =
//                        new OrientationEventListener(v.getContext())
{
//                            @Override
//                            public void onOrientationChanged(int
orientation) {
//                                int rotation;
//
//                                // Monitors orientation values to
determine the target rotation value
//                                if (orientation >= 45 && orientation
< 135) {
//                                    rotation = Surface.ROTATION_270;
//                                } else if (orientation >= 135 &&
orientation < 225) {
//                                    rotation = Surface.ROTATION_180;
//                                } else if (orientation >= 225 &&
orientation < 315) {
//                                    rotation = Surface.ROTATION_90;
//                                } else {
//                                    rotation = Surface.ROTATION_0;
//                                }
//
//
imageCapture.setTargetRotation(rotation);
//                            }
//                        };
//
//                orientationEventListener.enable();

                if(tipoCamera == 1) {
                    // Choose the camera by requiring a lens facing
                    cameraSelector = new CameraSelector.Builder()

.requireLensFacing(CameraSelector.LENS_FACING_FRONT)
                            .build();
                } else if(tipoCamera == 2){
                    // Choose the camera by requiring a lens facing
                    cameraSelector = new CameraSelector.Builder()

.requireLensFacing(CameraSelector.LENS_FACING_BACK)
                            .build();
                }

                // Attach use cases to the camera with the same
```

```
lifecycle owner
                camera = cameraProvider.bindToLifecycle(
                        ((LifecycleOwner) this),
                        cameraSelector,
                        preview,
                        imageCapture);

                // Connect the preview use case to the previewView

preview.setSurfaceProvider(previewView.getSurfaceProvider());
            } catch (InterruptedException | ExecutionException e) {
                // Currently no exceptions thrown.
cameraProviderFuture.get()
                // shouldn't block since the listener is being called,
so no need to
                // handle InterruptedException.
            }
        }, ContextCompat.getMainExecutor(v.getContext()));
    }

    private boolean allPermissionsGranted(View v) {
        if (ContextCompat.checkSelfPermission(
                v.getContext(), Manifest.permission.CAMERA) ==
PackageManager.PERMISSION_GRANTED) {
            return true;
        }
        return false;
    }

    @Override
    public View onCreateView(@NonNull LayoutInflater inflater,
@Nullable ViewGroup container,
                             @Nullable Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.fragment_pessoa2,
container, false);

        previewView = view.findViewById(R.id.viewFinder);
        ivFoto = view.findViewById(R.id.ivFoto);
        botaoGaleria = view.findViewById(R.id.btn_galeria);
        botaoTirarFoto = view.findViewById(R.id.btn_tirar_foto);
        botaoConfirmar = view.findViewById(R.id.btn_confirma_foto);
        botaoCancelar = view.findViewById(R.id.btn_cancela_foto);
        botaoProximo = view.findViewById(R.id.btn_continua_pessoa2);
        botaoTrocarCamera = view.findViewById(R.id.btn_troca_camera);

        // Request camera permissions
        if (allPermissionsGranted(view)) {

        } else {
            permissoesAcesso();
        }

        startCamera(view, tipoCamera);

        botaoProximo.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                ContentValues valores = new ContentValues();
                valores.put("nome", nome);
                valores.put("email", email);
                valores.put("telefone", telefone);
```

```
                valores.put("tipo_pessoa", tipoPessoa);
            }
        });
        return view;
    }

    @Override
    public void onActivityCreated(@Nullable Bundle savedInstanceState)
{
        super.onActivityCreated(savedInstanceState);
        mViewModel = new
ViewModelProvider(this).get(Pessoa2ViewModel.class);
        // TODO: Use the ViewModel

        // Preparar botões de Galeria e Câmera
        //Botão para editar foto do produto
        botaoGaleria.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Snackbar.make(v, "Acessando Galeria!",
Snackbar.LENGTH_LONG)
                        .setAction("Action", null).show();
                // Acesso à Galeria
                acessarGaleria();
                if(ivFoto != null){
                    ivFoto.setVisibility(View.VISIBLE);
                    previewView.setVisibility(View.INVISIBLE);
                    botaoGaleria.setVisibility(View.INVISIBLE);
                    botaoTirarFoto.setVisibility(View.INVISIBLE);
                    botaoConfirmar.setVisibility(View.VISIBLE);
                    botaoCancelar.setVisibility(View.VISIBLE);
                }

            }
        });

        //Botão para acionamento da câmera
        botaoTirarFoto.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Snackbar.make(v, "Capturando Imagem!",
Snackbar.LENGTH_LONG)
                        .setAction("Action", null).show();
                //Chamada do método tirarFoto()
                try {
                    tirarFoto2(v);
                } catch (IOException e) {
                    e.printStackTrace();
                }
                if(ivFoto != null){
                    ivFoto.setVisibility(View.VISIBLE);
                    previewView.setVisibility(View.INVISIBLE);
                    botaoGaleria.setVisibility(View.INVISIBLE);
                    botaoTirarFoto.setVisibility(View.INVISIBLE);
                    botaoConfirmar.setVisibility(View.VISIBLE);
                    botaoCancelar.setVisibility(View.VISIBLE);
                }
            }
        });

        botaoConfirmar.setOnClickListener(new View.OnClickListener() {
```

```
            @Override
            public void onClick(View v) {
                botaoProximo.setFocusable(true);
                botaoProximo.setVisibility(View.VISIBLE);
            }
        });

        botaoCancelar.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                ivFoto.setVisibility(View.INVISIBLE);
                previewView.setVisibility(View.VISIBLE);
                botaoGaleria.setVisibility(View.VISIBLE);
                botaoTirarFoto.setVisibility(View.VISIBLE);
                botaoConfirmar.setVisibility(View.INVISIBLE);
                botaoCancelar.setVisibility(View.INVISIBLE);
                botaoProximo.setVisibility(View.INVISIBLE);
            }
        });

        botaoTrocarCamera.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                cameraProvider.unbindAll();

                if(tipoCamera == 1) {
                    tipoCamera = 2;
                    // Choose the camera by requiring a lens facing
                    startCamera(v, tipoCamera);
                } else if(tipoCamera == 2){
                    tipoCamera = 1;
                    // Choose the camera by requiring a lens facing
                    startCamera(v, tipoCamera);
                }

            }
        });
    }

    private void tirarFoto2(View v) throws IOException {
        // Get a stable reference of the modifiable image capture use
case
        //imageCapture = imageCapture ?: return

        // Create time-stamped output file to hold the image
        File photoFile = new File(
                outputDirectory, String.valueOf(createImageFile()));

        // Create output options object which contains file + metadata
        ImageCapture.OutputFileOptions outputOptions =
                new
ImageCapture.OutputFileOptions.Builder(photoFile).build();

        // Set up image capture listener, which is triggered after
photo has
        // been taken

        imageCapture.takePicture(outputOptions,
ContextCompat.getMainExecutor(v.getContext()),
                new ImageCapture.OnImageSavedCallback() {
```

```
                    @Override
                    public void
onImageSaved(ImageCapture.OutputFileResults outputFileResults) {
                        // insert your code here.
                        Uri savedUri = Uri.fromFile(photoFile);
//                        Snackbar.make(v, "Foto capturada " +
//                                savedUri.getPath() + "!",
Snackbar.LENGTH_LONG).show();
                        // Chamada do método de adição da imagem na
galeria
                        galleryAddPic(savedUri);
                        // Definição das dimensões da imagem
                        setPic();
                    }

                    @Override
                    public void onError(ImageCaptureException error) {
                        // insert your code here.
                        Snackbar.make(v, "Erro ao capturar foto " +
                                error.getMessage() + "!",
Snackbar.LENGTH_LONG).show();
                    }
                }
        );
    }

    //Método para acessar a galeria
    private void acessarGaleria() {
        Intent intentPegaFoto = new Intent(Intent.ACTION_PICK,
                MediaStore.Images.Media.INTERNAL_CONTENT_URI);
        startActivityForResult(intentPegaFoto, PEGA_FOTO);
    }

    private void permissoesAcesso() {
        //Condicional para controle de permissões
        // Verifica se há permissão para leitura de arquivos
        if (ContextCompat.checkSelfPermission(this.getContext(),
                android.Manifest.permission.READ_EXTERNAL_STORAGE) !=
                PackageManager.PERMISSION_GRANTED) {
            if
(ActivityCompat.shouldShowRequestPermissionRationale(getActivity(),

android.Manifest.permission.READ_EXTERNAL_STORAGE)) {
            } else {
                ActivityCompat.requestPermissions(getActivity(),
                        new
String[]{android.Manifest.permission.READ_EXTERNAL_STORAGE},
PERMISSAO_REQUEST);
            }
        }

        //Verifica se há permissões para escrita de arquivos
        if (ContextCompat.checkSelfPermission(this.getContext(),
                android.Manifest.permission.WRITE_EXTERNAL_STORAGE) !=
PackageManager.PERMISSION_GRANTED) {

            if
(ActivityCompat.shouldShowRequestPermissionRationale(getActivity(),

android.Manifest.permission.WRITE_EXTERNAL_STORAGE)) {
            } else {
```

```
                ActivityCompat.requestPermissions(getActivity(),
                        new
String[]{android.Manifest.permission.WRITE_EXTERNAL_STORAGE},
PERMISSAO_REQUEST);
            }
        }

        //Verifica se há permissões para escrita de arquivos
        if (ContextCompat.checkSelfPermission(this.getContext(),
                Manifest.permission.CAMERA) !=
PackageManager.PERMISSION_GRANTED) {

            if
(ActivityCompat.shouldShowRequestPermissionRationale(getActivity(),
                    android.Manifest.permission.CAMERA)) {
            } else {
                ActivityCompat.requestPermissions(getActivity(),
                        new
String[]{android.Manifest.permission.CAMERA}, PERMISSAO_REQUEST);
            }
        }
    }

    private File createImageFile() throws IOException {
        // Create an image file name
        String timeStamp = new
SimpleDateFormat("yyyyMMdd_HHmmss").format(new Date());
        String imageFileName = "JPEG_" + timeStamp + "_";
        File storageDir =
this.getContext().getExternalFilesDir(Environment.DIRECTORY_PICTURES);
        File image = File.createTempFile(
                imageFileName,  /* prefix */
                ".jpg",         /* suffix */
                storageDir      /* directory */
        );

        // Save a file: path for use with ACTION_VIEW intents
        currentPhotoPath = image.getAbsolutePath();
        return image;
    }


    // Método que retorna o resultado da chamada da câmera pela Intent
    @Override
    public void onActivityResult(int requestCode, int resultCode,
@Nullable Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        //imageView =
this.getActivity().findViewById(R.id.foto_detalhe_produto);
        if (requestCode == REQUEST_TAKE_PHOTO && resultCode ==
RESULT_OK) {
            // Chamada do método de adição da imagem na galeria
            galleryAddPic(photoURI);
            // Definição das dimensões da imagem
            setPic();
        } else if (requestCode == PEGA_FOTO && resultCode ==
RESULT_OK) {
            //Captura caminho da imagem selecionada
            Uri imagemSelecionada = data.getData();

            // declara um stream (seguimento de dados) para ler a
```

```
imagem
            // recuperada do SD Card
            InputStream inputStream = null;

            // recuperando a sequencia de entrada, baseada no caminho
(uri)
            // da imagem
            try {
                inputStream = this.getActivity().getContentResolver()
                        .openInputStream(imagemSelecionada);
            } catch (FileNotFoundException e) {
                e.printStackTrace();
            }

            // recuperando um bitmap do stream
            bitmap = BitmapFactory.decodeStream(inputStream);
            // Vínculo do objeto ImageView
            //imageView = (AppCompatImageView) this.getActivity()
            //        .findViewById(R.id.foto_detalhe_produto);
            // Reduz imagem e configura apresentação
            Bitmap bitmapReduzido = Bitmap
                    .createScaledBitmap(bitmap, 1080, 1080, true);
            ivFoto.setImageBitmap(bitmapReduzido);
            ivFoto.setScaleType(AppCompatImageView.ScaleType.FIT_XY);
        }
    }

    //Método para inclusão de imagem na galeria
    private void galleryAddPic(Uri photoURI) {
        //File f = new File(currentPhotoPath);
        this.getActivity().sendBroadcast(
                new Intent(Intent.ACTION_MEDIA_SCANNER_SCAN_FILE,
                        photoURI));
    }

    //Método para definição das dimensões da imagem
    private void setPic() {
        // Obtendo as dimensões da imagem para a View
        int targetW = ivFoto.getWidth();
        int targetH = ivFoto.getHeight();

        // Obter as dimensões do Bitmap
        BitmapFactory.Options bmOptions = new BitmapFactory.Options();
        //bmOptions.inJustDecodeBounds = true;
        //BitmapFactory.decodeFile(currentPhotoPath, bmOptions);
        int photoW = bmOptions.outWidth;
        int photoH = bmOptions.outHeight;
        // Determina como diminuir a escala da imagem
        int scaleFactor = Math.min(photoW / targetW, photoH /
targetH);

        // Decodifica o arquivo de imagem para o Bitmap que preencherá
a View
        bmOptions.inJustDecodeBounds = false;
        bmOptions.inSampleSize = scaleFactor;
        bmOptions.inPurgeable = true;

        // Cria o bitmap da imagem capturada
        bitmap = BitmapFactory.decodeFile(currentPhotoPath,
bmOptions);
```

```
        // Declara objeto vetor e rotaciona em 90 graus a imagem
        Matrix matrix = new Matrix();
        matrix.postRotate(270);
        if(tipoCamera == 2){
            matrix.invert(matrix);
        }
        //matrix.preRotate(45);
        Bitmap rotated = Bitmap.createBitmap(bitmap, 0, 0,
bitmap.getWidth(),
                bitmap.getHeight(), matrix, true);
        // Devolve imagem para o ImageView
        ivFoto.setImageBitmap(rotated);

        // Apresenta a imagem na tela
        //ivFoto.setImageBitmap(bitmap);
    }

    // TODO -
    public boolean isExternalStorageWritable() {
        String state = Environment.getExternalStorageState();
        if (Environment.MEDIA_MOUNTED.equals(state)) {
            return true;
        }
        return false;
    }

    // externo
    public File getAlbumStorageDir(String albumName) {
        // Get the directory for the user's public pictures directory.
        File file = new
File(Environment.getExternalStoragePublicDirectory(
                Environment.DIRECTORY_DCIM), albumName);
        if (!file.mkdirs()) {
            Log.e(TAG, "Directory not created");
        }
        return file;
    }

    // local
    public File getAlbumStorageDir(Context context, String albumName)
{
        // Get the directory for the app's private pictures directory.
        File file = new File(context.getExternalFilesDir(
                Environment.DIRECTORY_DCIM), albumName);
        if (!file.mkdirs()) {
            Log.e(TAG, "Directory not created");
        }
        return file;
    }
}
```

Referência Bibliográfica

[1] https://developer.android.com/training/camerax. Acessado em 04/07/2021.