```
package es.uma.processimage;

import android.Manifest;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.BatteryManager;
import android.os.Bundle;
import android.os.Environment;
import android.provider.MediaStore;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v4.content.FileProvider;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Spinner;
import android.widget.Toast;

import java.io.File;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;



public class MainActivity extends AppCompatActivity {

    private Button button;
    private Button buttonProc;
    private ImageView iv1;
    private static ImageView iv2;

    public static final int REQUEST_IMAGE = 100;
    public static final int REQUEST_PERMISSION = 200;
    private static String imageFilePath = "";
    static int  optionSelected = 0;
    static Context c;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        button = findViewById(R.id.button);
        buttonProc = findViewById(R.id.buttonProcess);
        iv1 = findViewById(R.id.image1);
        iv2 = findViewById(R.id.image2);
        c=this;

        if (ContextCompat.checkSelfPermission(this, Manifest.permission.WRITE_EXTERNAL_STORAGE) !=
                PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(this, new String[] {Manifest.permission.WRITE_EXTERNAL_STORAGE},
                    REQUEST_PERMISSION);
        }

        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                openCameraIntent();
            }
        });
        buttonProc.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                myAsyncTask at = new myAsyncTask();
                at.execute();
            }
        });
        Spinner spinner = (Spinner) findViewById(R.id.spinner);
        spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
            @Override
            public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {
                optionSelected = position;
                Log.i("Rafa", "optionSelected = " + optionSelected);
            }

            @Override
            public void onNothingSelected(AdapterView<?> parent) {

            }
        });

    }

    private void openCameraIntent() {
        Intent pictureIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
        if (pictureIntent.resolveActivity(getPackageManager()) != null) {

            File photoFile = null;
            try {
                photoFile = createImageFile();
            }
            catch (IOException e) {
                e.printStackTrace();
                return;
            }
            Uri photoUri = FileProvider.getUriForFile(this, getPackageName() +".provider", photoFile);
            pictureIntent.putExtra(MediaStore.EXTRA_OUTPUT, photoUri);
            startActivityForResult(pictureIntent, REQUEST_IMAGE);
        }
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);

        if (requestCode == REQUEST_PERMISSION && grantResults.length > 0) {
            if (grantResults[0] == PackageManager.PERMISSION_GRANTED) {
                Toast.makeText(this, "Thanks for granting Permission", Toast.LENGTH_SHORT).show();
            }
```

```
        }
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        if (requestCode == REQUEST_IMAGE) {
            if (resultCode == RESULT_OK) {
                iv1.setImageURI(Uri.parse(imageFilePath));
                Log.i("Rafa","File name: "+imageFilePath);

            }
            else if (resultCode == RESULT_CANCELED) {
                Toast.makeText(this, "You cancelled the operation", Toast.LENGTH_SHORT).show
();
            }
        }
    }

    private static class myAsyncTask extends AsyncTask<Void,Void,Bitmap> {
        @Override
        protected Bitmap doInBackground(Void... v) {
            return  processImage();
        }

        @Override
        protected void onPostExecute(Bitmap bMap) {
            if(bMap!=null) iv2.setImageBitmap(bMap);
            else Toast.makeText(c,"There is not picture!",Toast.LENGTH_SHORT).show();
        }
    }

    public static Bitmap processImage(){
        if(!imageFilePath.matches("")) {
            BitmapFactory.Options options = new BitmapFactory.Options();
            options.inMutable = true;
            Bitmap bMap = BitmapFactory.decodeFile(imageFilePath, options);
            Log.i("Rafa", "Bitmap config= " + bMap.getConfig());
            Log.i("Rafa", "Bitmap density= " + bMap.getDensity());
            Log.i("Rafa", "Bitmap size= " + bMap.getHeight() + " x " + bMap.getWidth());
            FiltroImagen st = new FiltroAzul();
            switch (optionSelected) {
                case 0:
                    st = new FiltroAzul();
                    break;
                case 1:
                    st = new FiltroAzulPorRojo();
                    break;
                case 2:
                    st = new FiltroStereograma();
                    break;
                case 3:
                    st = new FiltroMedia();
                    break;
                case 4:
                    st = FiltroMatriz.creaFiltroBordes();
                    break;
                case 5:
                    st = FiltroMatriz.creaFiltroEnfoque();
                    break;
                case 6:
                    st = FiltroMatriz.creaFiltroMedia();
                    break;
                case 7:
                    st = FiltroMatriz.creaFiltroRelieve();
                    break;
            }
            //Bitmap newbMap = bMap.copy(Bitmap.Config.ARGB_8888,true);
            st.filtra(bMap);
            return bMap;
        }
        return null;
    }

    private File createImageFile() throws IOException{

        String timeStamp = new SimpleDateFormat("yyyyMMdd_HHmmss", Locale.getDefault()).form
at(new Date());
        String imageFileName = "IMG_" + timeStamp + "_";
        File storageDir = getExternalFilesDir(Environment.DIRECTORY_PICTURES);
        //createTempFile aÃ±ade un string random entre imageFileName y .jpg
        File image = File.createTempFile(imageFileName, ".jpg", storageDir);
        imageFilePath = image.getAbsolutePath();

        return image;
    }

}
```