```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "provenance": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "code",
      "execution_count": 1,
      "metadata": {
        "id": "M9TDYjHFMfBL"
      },
      "outputs": [],
      "source": [
        "import keras\n",
        "from keras.preprocessing.image import ImageDataGenerator"
      ]
    },
    {
      "cell_type": "code",
      "source": [
        "#Define the parameters/arguments for ImageDataGenerator class\n",
        "train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,rotation
_range=180,zoom_range=0.2,horizontal_flip=True)\n",
        "\n",
        "test_datagen=ImageDataGenerator(rescale=1./255)"
      ],
      "metadata": {
        "id": "KEFqxyqwMqMu"
      },
      "execution_count": 2,
      "outputs": []
    },
    {
      "cell_type": "code",
      "source": [
        "#Applying ImageDataGenerator functionality to trainset\n",
        "x_train=train_datagen.flow_from_directory('/content/drive/MyDrive/Dataset/Dat
aset/train_set',target_size=(128,128),batch_size=32,class_mode='binary')"
      ],
```

```
    "metadata": {
     "colab": {
      "base_uri": "https://localhost:8080/"
     },
     "id": "D5S4LCT-Mu2L",
     "outputId": "a4113be0-c0b3-4a95-bf3e-ea77de8b744d"
    },
    "execution_count": 4,
    "outputs": [
     {
      "output_type": "stream",
      "name": "stdout",
      "text": [
       "Found 436 images belonging to 2 classes.\n"
      ]
     }
    ]
   },
   {
    "cell_type": "code",
    "source": [
     "#Applying ImageDataGenerator functionality to testset\n",
     "x_test=test_datagen.flow_from_directory('/content/drive/MyDrive/Dataset/Data
set/test_set',target_size=(128,128),batch_size=32,class_mode='binary')"
    ],
    "metadata": {
     "colab": {
      "base_uri": "https://localhost:8080/"
     },
     "id": "rQakYllkMxQG",
     "outputId": "5d4f7c20-34c2-42ca-e0d8-a8fbfedf01c4"
    },
    "execution_count": 5,
    "outputs": [
     {
      "output_type": "stream",
      "name": "stdout",
      "text": [
       "Found 121 images belonging to 2 classes.\n"
      ]
     }
    ]
   },
   {
    "cell_type": "code",
    "source": [
     "#import model building libraries\n",
     "\n",
     "#To define Linear initialisation import Sequential\n",
     "from keras.models import Sequential\n",
```

```
        "#To add layers import Dense\n",
        "from keras.layers import Dense\n",
        "#To create Convolution kernel import Convolution2D\n",
        "from keras.layers import Convolution2D\n",
        "#import Maxpooling layer\n",
        "from keras.layers import MaxPooling2D\n",
        "#import flatten layer\n",
        "from keras.layers import Flatten\n",
        "import warnings\n",
        "warnings.filterwarnings('ignore')"
       ],
       "metadata": {
        "id": "tbh9PKR1M9H8"
       },
       "execution_count": 6,
       "outputs": []
      },
      {
       "cell_type": "code",
       "source": [
        "#initializing the model\n",
        "model=Sequential()"
       ],
       "metadata": {
        "id": "vlNVG-FhM_gF"
       },
       "execution_count": 7,
       "outputs": []
      },
      {
       "cell_type": "code",
       "source": [
        "#add convolutional layer\n",
        "model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))\
n",
        "#add maxpooling layer\n",
        "model.add(MaxPooling2D(pool_size=(2,2)))\n",
        "#add flatten layer \n",
        "model.add(Flatten())"
       ],
       "metadata": {
        "id": "Ul-pbzGNNCQ5"
       },
       "execution_count": 8,
       "outputs": []
      },
      {
       "cell_type": "code",
       "source": [
        "#add hidden layer\n",
```

```
        "model.add(Dense(150,activation='relu'))\n",
        "#add output layer\n",
        "model.add(Dense(1,activation='sigmoid'))"
       ],
       "metadata": {
        "id": "CHkc6tNmND_Q"
       },
       "execution_count": 10,
       "outputs": []
     },
     {
       "cell_type": "code",
       "source": [
        "#configure the learning process\n",
        "model.compile(loss='binary_crossentropy',optimizer=\"adam\",metrics=[\"accur
acy\"])"
       ],
       "metadata": {
        "id": "T71QZpAVNH7P"
       },
       "execution_count": 11,
       "outputs": []
     },
     {
       "cell_type": "code",
       "source": [],
       "metadata": {
        "id": "aTbX1IgCN8Md"
       },
       "execution_count": null,
       "outputs": []
     }
   ]
}
```