

Documentação: Image Capture, Firebase RealTime Database and Storage

(captura de imagem, firebase em tempo real e armazenamento)

Objetivo:

Capturar imagem automaticamente através de uma câmera, armazenar as imagens e mostrar suas devidas informações e especificações em tempo real utilizando a plataforma Firebase do google.

Linguagem de programação usada:

- Python

Importações, Pacotes e Coleções usadas no programa:

```
import cv2
import time
import datetime as date
import os
from threading import Thread
import multiprocessing
```

```
import pyrebase
```

Ao todo utilizamos as seguintes importações acima.

cv2: biblioteca que possui módulos de processamentos de imagem.

time: este módulo fornece várias funções relacionadas ao tempo.

datetime: este módulo fornece classes para manipular datas e horas de maneira simples e complexa.

os: Este módulo fornece uma maneira portátil de usar a funcionalidade dependente do sistema operacional, ou seja, manipular caminhos.

Thread: importa varias funções dentre elas a função de tempo

multiprocessing: é um pacote que suporta processos e oferece simultaneidade local e remota

pyrebase: é uma biblioteca em Python utilizada para criação de aplicações com conexão com API Google Firebase.

Código:

1º Método: Captura de imagem

```
import cv2
import time
import datetime as date
import os
from threading import Thread
import multiprocessing

class image_capture:
    def __init__(self):
        self.__key=True
    def take_picture(self,path):
        cam = cv2.VideoCapture(0 + cv2.CAP_DSHOW)
        frame = cam.read()[1]
        cv2.imwrite(filename=(path),img=frame)

    def __file_comparator(self,path):
        if not os.path.exists(path):
            os.makedirs(path)
    def __program_finisher(self):
        while True:
            key=input("Write stop if you want to end the program \n")
            if(key=="stop"):
                self.__key=False
                break
```

Iniciamos essa primeira classe com a captura de imagens, com 1º função usando a biblioteca cv2 e alocando cada imagem tirada da câmera a uma pasta, a 2º função compara se há alguma pasta já criada caso não exista uma nova é criada do zero, já a 3º função ele permite que o programa continue rodando até que o usuário digite a chave “stop”, assim que digitado o programa para imediatamente.

```

def multiple_pictures(self, delay, initial_path):
    start = time.time()

    t1 = Thread(target=self.__program_finisher, args=())
    t1.start()
    while self.__key:
        path = initial_path
        path_date = date.datetime.date(date.datetime.now())
        path=path+str(path_date.year)
        self.__file_comparator(path)

        path=path+'/'
        path = path+str(path_date.month)
        self.__file_comparator(path)

        path=path+'/'
        path = path+str(path_date.day)
        self.__file_comparator(path)

        date_hour = date.datetime.time(date.datetime.now())
        hour = str(date_hour.hour)+"_"+str(date_hour.minute)+"_"+str(date_hour.second)
        self.take_picture((path + "/" + hour + ".jpg"))
        time.sleep(delay+1)

```

Nesta função o programa irá capturar múltiplas imagens, isso enquanto ele organizar o local em que vai ser armazenado as imagens capturadas, sendo a primeira pasta o ano, a segunda pasta o mês e a terceira pasta o dia e com as informações de hora, minutos e segundos que foi capturada a imagem.

2º Método: Salvamento das imagens e informações

```
import pyrebase

class storagePyreBase():
    def __init__(self):
        config = {
            "apiKey": "AIzaSyAgWzJypcKysr7Negw6mhNbcy3pZOKmfSA",
            "authDomain": "imageai-2018.firebaseio.com",
            "databaseURL": "https://imageai-2018.firebaseio.com",
            "projectId": "imageai-2018",
            "storageBucket": "imageai-2018.appspot.com",
            "messagingSenderId": "862027968333"
        }

        firebase = pyrebase.initialize_app(config)

        auth = firebase.auth()

        self.__users = auth.sign_in_with_email_and_password('Insira seu email aqui', 'senha')
        self.__db = firebase.database()
        self.__storage = firebase.storage();
```

Nessa segunda classe importamos a biblioteca pyrebase e iniciamos a função com uma chave de configuração para ter conexão com a base de dados do firebase, a biblioteca é instanciada para inicializar a conexão com dados. Em seguida é feita a autenticação para autorizar a configuração, depois para que a autenticação seja completa e sem falha, é necessário que o usuário entre com uma conta no firebase e a cadastre para que todas as alterações e autenticações sejam feitas somente para os usuários cadastrados, tornando o database e o storage privados.

```

def saveArquivo (self, enderecoFirebase, enderecoArquivo):
    print(enderecoFirebase)
    results = self.__storage.child(enderecoFirebase).put(enderecoArquivo, self.__users['idToken'])

def salvarDados(self, json_s):
    import pyrebase

    data = [{"Local": "Lapin 2018",
    "mes": "Dezembro",
    "Dia": 10,
    "Status da camera": "Off" },
    {"Local": "CAMÊRA",
    "Fotos ": "13_12_18",
    "Local de armazenamento": "Raspberry",
    "Status": "informações das fotos"}]

    for elemento in data:
        nome = elemento['Local']
        self.__db.child("PROJECT IMAGEAI").child(nome).set(elemento, self.__users['idToken'])

```

E por fim terminamos com as duas ultimas funções, sendo a 1º para salvar as imagens e onde será o endereço no firebase e de onde ele ira pegar o endereço da imagem. A 2º função consiste em mostrar as informações quando as imagens estiverem no storage, e logo é dado o titulo da imagem para ser observada no sotrage, para que todos os envios de dados sejam feitos corretamente é necessário que os usuários tenha que da a permissão para o programa e a base de dados firebase possam fazer isso, para isso é usado o 'idToken'.

3º Método: Terminal e execução

```

from image_capture import image_capture as ic

path=input("Diga o caminho sem /\n")
path=path+'/'
a = ic()
a.multiple_pictures(3,path)

```

A ultima classe é somente a execução do programa, onde ira abrir um terminal e perguntar qual o caminho que as imagens devem ir e também a última linha do código representa de quantos em quantos segundos a câmera irá capturar imagens.

Conceito do programa:

O programa deve capturar imagens em tempo real de n em n segundos (definidos pelo usuário) e suas devidas imagens e informações devem ser armazenadas no Firebase, sendo as informações e especificações no realtime database e a imagem no storage do firebase. O programa usa o treinamento de imagens oferecidos pelo opencv2 em que ele fará o reconhecimento dos objetos capturados pela câmera e mostra a probabilidade de acerto de ser aquele objeto.

O programa cria uma pasta no computador em que ficará armazenadas as imagens que forem capturadas pela câmera enquanto essas imagens também são enviadas para o firebase com um pequeno delay, as informações são feitas pelo json que é uma sintaxe para carregamento de dados.

Ao final poderá ser checado no firebase e no seu database que todas as informações e imagens estarão armazenadas nas nuvens, tudo isso em tempo real.

Amostra do resultado do programa em tempo real:











Storage

FILES

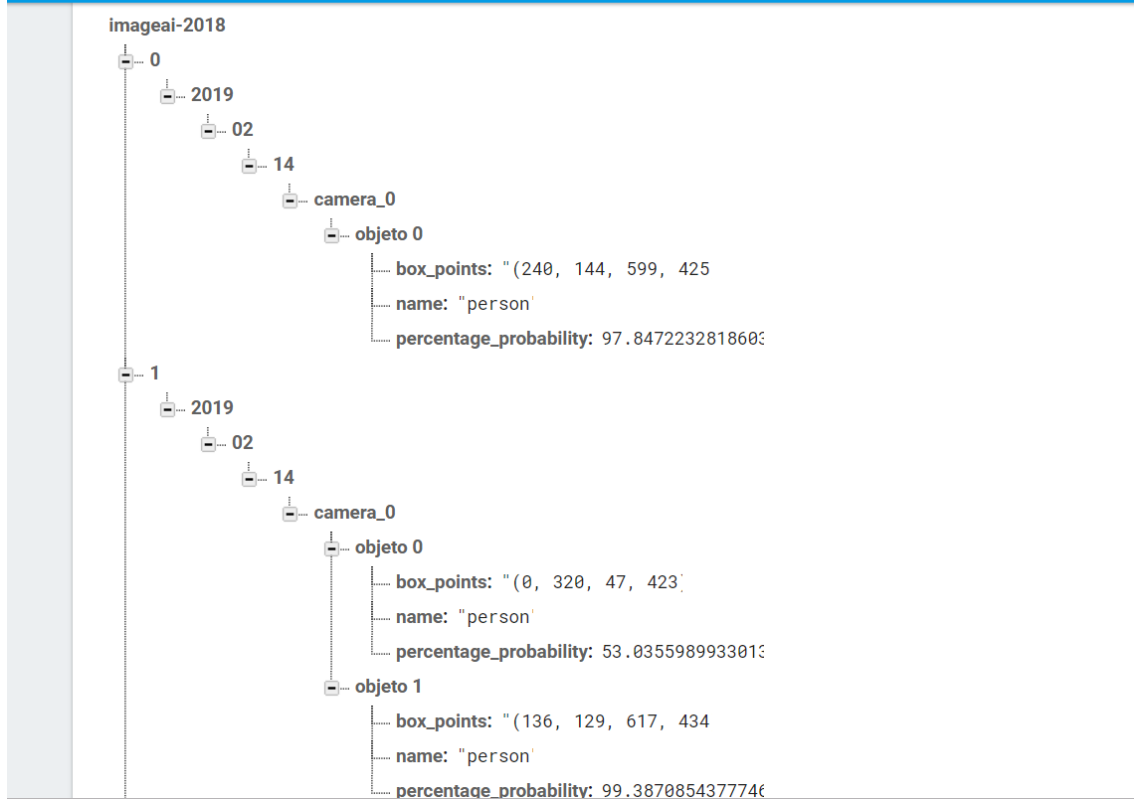
RULES

gs://cropped-images.appspot.com > images

UPLOAD FILE

<input type="checkbox"/>	Name	Size	Type	Last modified
<input type="checkbox"/>	 1d7ce39bef9b21a0c9f6dd30f6e43e44.jpg	28.02 KB	image/jpeg	—
<input type="checkbox"/>	 1d7ce39bef9b21a0c9f6dd30f6e43e44Cropp...	13.11 KB	image/jpeg	—
<input type="checkbox"/>	 2d667c1606e23cabe4c7ccd442c2aa2a.jpg	59.71 KB	image/jpeg	—
<input type="checkbox"/>	 2d667c1606e23cabe4c7ccd442c2aa2aCrop...	30.49 KB	image/jpeg	—
<input type="checkbox"/>	 64fe9f9e62e628ef4a949339d9ac63de.jpg	111.05 KB	image/jpeg	—
<input type="checkbox"/>	 64fe9f9e62e628ef4a949339d9ac63deCropp...	24.84 KB	image/jpeg	—
<input type="checkbox"/>	 e39e37a9ad04ffb0a653f966d385a65fCropp...	33.16 KB	image/jpeg	—
<input type="checkbox"/>	 f8c5c821649aa1a9a49f9e6825b1dd85.jpg	41.78 KB	image/jpeg	—
<input type="checkbox"/>	 f8c5c821649aa1a9a49f9e6825b1dd85Crop...	22.52 KB	image/jpeg	—
<input type="checkbox"/>	 Tang.png	118.05 KB	image/png	—

Amostra de imagens armazenadas no storage.



Dados de imagens armazenadas no database.