Biz bu dasturimizni yaratishda Python dasturlash tilidan foydalanamiz. Shuning uchun kompyuterimizda Python dasturlash tili o‘rnatilgan bo‘lishi kerak. Buning uchun internet brouzeriga *python.org* deb yozib rasmiy saytidan yuklab, o‘rnatib olamiz.

**Kerakli kutubxonalarni o‘rnatish.**

Kompyterdagi **command prompt** dasturidan foydalanamiz. Dastur oynasi ochilgandan so‘ng `python` deb yozib Enter tugmasini bosamiz. Keyin `pip install face-recognition` deb yozib Enterni bosamiz. Shu tartibda Numpy kutubxonasini o‘rnatish uchun `pip install numpy`, OpenCV kutubxonasini o‘rnatish uchun esa, `pip install opencv-python` deb yozamiz.

**Kerakli kutubxonalarni chaqirib olish .**

Kutubxonalarni chaqirib olish uchun `import` operatoridan foydalanamiz.

```
import cv2
```

Bu OpenCV kutubxona kompyuterni ko‘rishi bilan ishlaydigan eng katta kutubxonasi hisoblanadi. OpenCV (Open Source Computer Vision Library) ochiq manbali kompyuter ko‘rish va mashinani o‘rganish dasturlari kutubxonasi. Kutubxonada 2500 dan ortiq optimallashtirilgan algoritmlar mavjud bo‘lib, ular klassik va zamonaviy kompyuter ko‘rish va mashinani o‘rganish algoritmlarining keng qamrovli to‘plamini o‘z ichiga oladi. Ushbu algoritmlar yuzlarni aniqlash va tanib olish, ob'ektlarni aniqlash, videolarda inson harakatlarini tasniflash, kamera harakatlarini kuzatish, harakatlanuvchi ob'ektlarni kuzatish, ob'ektlarning 3D modellarini ajratib olish, stereo kameralardan 3D nuqta bulutlarini ishlab chiqarish, yuqori piksellar sonini yaratish uchun tasvirlarni birlashtirish uchun ishlatilishi mumkin.

```
import numpy as np
```

NumPy Pythonda ilmiy hisoblashlar uchun asosiy paketdir, ya’ni matematik hisob kitoblar bilan ishlovchi kutubxonasidir. Biz numpyni foydalanishga qulay bo‘lishi uchun np deb belgilab oldik.

```
import face_recognition
```

`face_recognition` (Yuzni tanib olish) - bu ularning fotosuratlari va videolari asosida yuzlarni tanib olish jarayoni yoki usuli bo‘lib, bu tizimlar ayniqsa huquqni muhofaza qilish organlari va boshqa sohalarda keng qo‘llaniladi. Bu kutubxona loyihamizni asosini tashkil etadi.

```
import os
```

Ushbu modul operatsion tizimga bog'liq funksiyalardan foydalanishning ko‘chma usulini taqdim etadi. Bu moduldan papkadan fayllarni chaqirib oldim.

```
from datetime import datetime
```

Python datetime - bu pythonda sanalar bilan ishlash moduli bo‘lib, unda sana va vaqt operatsiyalari uchun asosan 4 ta asosiy ob'ekt mavjud: sana, vaqt, sana va timedelta.

**Train jarayoniga ma’lumotlarni tayyorlash**

```
path = "train"
images = []
className = []
myList = os.listdir(path)
print(myList)
```

`path` datasetimizni yuklab olamiz. Ya’ni o‘sha datasetda bor bo‘lgan yuzlarni taniydi, va faylga yozib qo‘yadi.

`images` o‘zgaruvchisi datasetdagi rasmlarni saqlab oladi.

`classNamega` faqat rasmlarni nomini ya’ni .jpg formatisiz yuklaymiz.

`myList` o‘zgaruvchisiga datasetdagi rasmlarni to‘liq nomini oladi.

`Print` operatori orqali `myList` o‘zgaruvchisi qiymatlarini chop etamiz.

Natija:

```
['Muminov_Baxtiyor.jpg','Tursunov_Azim.jpg',
'Ahmadjonov_Muhammadjon.jpg', 'Ikromov_Aslbek.jpg', . .
. . Ayapberginov_Raul.jpg']
```

**Fayldan ismlarni ajratib olish.**

```
for cl in myList:
  curImg = cv2.imread(f"{path}/{cl}")
  images.append(curImg)
  className.append(os.path.splitext(cl)[0])
print(className)
```

Bunda for takrorlanish operatoridan foydalanamiz. `myList` ichidagi har bir rasmni olib, `cv2.imread` yordamida rasmni o‘qib rasmni matretsa ko‘rinishiga o‘tkazib `curImg` o‘zgaruvchisiga yuklab qo‘yamiz.

`images` o‘zgaruvchisiga `append` yordamida qo‘shib qo‘yamiz.

`className` o‘zgaruvchisiga rasmlarni nomini yuklab qo‘yamiz. `className` ma’lumot turi ro‘yxat.

`print` orqali `className` qiymatini chop etamiz.

Natija:

```
['Muminov_Baxtiyor', 'Tursunov_Azim',
'Ahmadjonov_Muhammadjon', . . . . 'Ikromov_Aslbek',
```

Model o‘qitish va natija

```
def findEncodings(image):
  encodeList = []
  for img in images:
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    encode = face_recognition.face_encodings(img)[0
]
    encodeList.append(encode)
  return encodeList
```

`findEncodings` funksiyasi datasetdagi rasmlarni kodlashtirib beradi.

`encodeList` o‘zgaruvchisiga kodlashtirilgan rasmlarni ro‘yxat ko‘rinishida saqlaymiz.

`for` yordamida `images` o‘zgaruvchisiga yuklab qo‘ygan rasmlarnimizni birin ketin `img` o‘zgaruvchisiga yuklaymiz.

`img` o‘zgaruvchisidagi rasm BGR formatidagi rangda bo‘ladi, shuni `cv2.cvtColor()` funksiyasi yordamida, RGB formatiga o‘tkazib yana `img` o‘zgaruvchisiga yuklab qo‘yamiz.

`encode` o‘zgaruvchiga `face_recognition.face_encodings()` funksiyasi yordamida `img` o‘zgaruvchisidagi yuz rasmini kodlashtirib yuklaymiz.

Kodlashtirilgan yuzni har safar `encodeList` o‘zgaruvchisiga `append` funksiyasi yordamida qo‘shib ketaveramiz, va oxirida funksiyadan `encodeList` ni `return(qaytaramiz)`.

```
def markAttendance(name):
  info = ["AB-225","4"]
  student = [
      "Saparov_Sunnat",
```

```
        "Juraqulov_Ibratjon",
        "Muminov_Baxtiyor",
        "Tojiyev_Sevdiyor",
          .
          .
        "Tuxtashev_Shohruh",
        "Ahmadjonov_Muhammadjon",
        "Ayapberginov_Raul",
        "Ibragimov_Dilshod"]
    summary = dict.fromkeys(student,info)
```

`markAttendance` funksiyasi datasetimizda bor bo‘lgan yuzni kamerada ko‘rganda, faylga talaba ismini, guruhini, kursini va kameraga tushgan vaqtini yozib qo‘yadi.

`info` o‘zgaruvchiga talabaning guruhi va kursi ro‘yxat ko‘rinishida yuklangan.

`student` o‘zgaruvchisiga talabaning familiyasi va ismi ro‘yxat ko‘rinishida yuklangan.

`summary` o‘zgaruvchisiga `dict.fromkeys()` funksiyasi yordamida `student` va `info` o‘zgaruvchilarini lug’at ma’lumot turiga o‘tkazayapman. Bunda `student` kalit `info` esa qiymat bo‘layapti.

```
 with open("information.csv","r+") as f:
    nameList = []
    myDataList = f.readline()
    for line in myDataList:
      entry = line.split(',')
      nameList.append(entry[0])
    if name not in nameList:
      now = datetime.now()
      dtString = now.strftime('%H:%M:%S')
      f.writelines(f"\n{name},{','.join(summary[name])}
,{dtString}")

encodeListKnown = findEncodings(images)
print('Encoding Complete')
```

`with open("information.csv","r+") as f:` bunda `information.csv` faylini o‘qish uchun ochdim va `f` deb belgilab oldim.

`nameList` o‘zgaruvchiga talabalar ism familiyasini ro‘yxat ma’lumot turida saqlaymiz.

`myDataList` o‘zgaruvchiga `f.readline()` funksiyasi yordamida fayl ichidagi ma’lumotni satr bo‘yicha o‘qib yuklaydi.

`for line in myDataList:` bunda `myDataList` o‘zgaruvchi ichidagi har bir satrni `line` o‘zgaruvchisiga berayapdi.

`entry` o‘zgaruvchisiga `split(',')` funksiyasi yordamida `line` o‘zgaruvchi ichidagi ma’lumotni satr ma’lumot turidan, ro‘yxat ma’lumot turiga o‘tkazayapdi.

`nameList` o‘zgaruvchiga `append()` funksiyasi yordamida `entry` ro‘yxat o‘zgaruvchisini nolinchi indeksini qo‘shdim, chunki nolinchi index talabaning ism familiyasi.

`if name not in nameList:` bo‘nda `name` ya’ni talabaning ism familiyasi `nameList` o‘zgaruvchisida bo‘lmasa, `if` shart operatori ishlaydi. Chunki bir talabani takroran yozmasligi kerak.

`now` o‘zgaruvchiga `datetime.now()` funksiyasi yordamida talaba kameraga tushgan vaqti yuklanadi.

`dtString` o‘zgaruvchiga `strftime('%H:%M:%S')` funksiyasi yordamida `now` o‘zgaruvchini qiymati soat, daqiqa, soniyaga o‘tkazib yukladim.

`f.writelines(f"\n{name},{','.join(summary[name])},{dtString}")` bunda `f` o‘zgaruvchidagi faylga talabaning ismi, guruhi, kursi va vaqtni yozdim.

`encodeListKnown` o‘zgaruvchiga `findEncodings()` funksiyasi yordamida `images` rasmlarni kodlab yukladim.

`print('Encoding Complete')` kodlash jarayoni nihoysiga yetganda `'Encoding Complete'` ya’ni ‘kodlash nihoyasiga yetdi’ yozuvi chop etiladi.

```
cap = cv2.VideoCapture(0)
```

`cap` o‘zgaruvchisiga `cv2.VideoCapture(0)` funksiyasi yordamida kamerani ishga tushirib ma’lumot olib yuklaydi. 0 qiymat esa kamerani ishga tushirishni bildiradi. Bundan tashqari simsiz kameralar yoki videodan ham foydalansa bo‘ladi.

```
while True:
  success, img = cap.read()
  #img = captureScreen()
  imgS = cv2.resize(img,(0,0),None,0.25,0.25)
  imgS = cv2.cvtColor(imgS, cv2.COLOR_BGR2RGB)

  facesCurFrame = face_recognition.face_locations(i
mgS)
  encodesCurFrame = face_recognition.face_encodings
(imgS,facesCurFrame)
```

`success, img = cap.read()` kameradan ma'lumotni matritsa ko'rinishida o'qib oladi .

`imgS = cv2.resize(img,(0,0),None,0.25,0.25)` `img` o'zgaruvchidagi matritsani o'lchmini o'zgartiradi. Bu rasmni kodlashtirish uchun qilinayapdi.

`imgS = cv2.cvtColor(imgS, cv2.COLOR_BGR2RGB)` rasmni rangini RGB ga o'tkazilayapdi.

`facesCurFrame = face_recognition.face_locations(imgS)` rasmdan yuzni kordinatalarini `face_recognition.face_locations()` funksiyasi yordamida aniqlab oldim.

`encodesCurFrame = face_recognition.face_encodings(imgS, facesCurFrame)` kameradan olinayotgan ma'lumotni kodlab beradi.

```
for encodeFace, faceLoc in zip(encodesCurFrame, fac
esCurFrame):
    matches = face_recognition.compare_faces(encode
ListKnown,encodeFace)
    faceDis = face_recognition.face_distance(encode
ListKnown,encodeFace)
    matchIndex = np.argmin(faceDis)
```

`for encodeFace, faceLoc in zip(encodesCurFrame, facesCurFrame):` bunda `encodesCutFrame` va `facesCurFrame` o'zgaruvchilarning har bir qiymatini `encodeFace` va `faceLoc` o'zgaruvchilariga yuklayapman.

`matches = face_recognition.compare_faces(encodeListKnown,encodeFace)` bunda `matches` o‘zgaruvchisiga datasetdagi har bir yuzlarning rasmi bilan kamera orqali olinayotgan rasmlarni solishtiradi buni `compare_faces()` funksiyasi yordamida qildim, agar kameradan o‘qilayotgan rasm bilan datasetdagi rasmni birortasiga mos kelsa, True qiymat yozadi aks holda False. `matches` o‘zgaruvchisinig ma’lumot turi ro‘yxat.

`faceDis = face_recognition.face_distance(encodeListKnown,encodeFace)` bunda `faceDis` o‘zgaruvchisiga datasetdagi har bir yuzlarning rasmi bilan kameradan orqali olinayotgan rasmlarni solishtiradi va kameradan o‘qilayotgan rasm bilan datasetdagi rasmnig birortasiga o‘xshashroq bo‘lsa, orasidagi farqning qiymati shunchalik kichik bo‘ladi, buni `face_distance()` funksiyasi yordamida qildim, agar kameradan o‘qilayotgan rasm bilan datasetdagi rasmni birortasiga mos kelmasa, orasidagi farq shuncha katta bo‘ladi. `faceDis` o‘zgaruvchisining ma’lumot turi ro‘yxat.

`matchIndex = np.argmin(faceDis)` bunda, `matchIndex` o‘zgaruvchisga `argmin()` funksiyasi yordamida `faceDis` o‘zgaruvchisining eng kichik qiymatni indeksini yuklayapman.

```
if matches[matchIndex]:
      name = className[matchIndex]
      print(name)
      y1,x2,y2,x1 = faceLoc
      y1,x2,y2,x1 = y1*4,x2*4,y2*4,x1*4
      #cv2.rectangle(img, (x1,y1),(x2,y2),(0,255,0),2)
      #cv2.rectangle(img, (x2,y2-
35),(x2,y2),(0,255,0),cv2.FILLED)
      #cv2.putText(imgS,name,(x1,y2),cv2.FONT_HERSHEY_C
OMPLEX,1,(255,255,255),2)
      markAttendance(name)
  cv2_imshow(imgS)
  cv2.waitKey(0)
```

`if matches[matchIndex]:` bu shart operatirida faqatgina shart `True` qaytarsagina ishlaydi. Shunday ekan `matches` o‘zgaruvchisini `matchIndex` indeksi oladi va `True` qaytaradi.

`name = className[matchIndex]` bunda `name` o‘zgaruvchisiga `className`(datasetdagi rasmlarni ismlari saqlangan ro‘yxat ma’lumot turilik)

o‘zgaruvchisi ichidagi `matchIndex`(tanib olingan rasmning indeksi) indexsini yuklaydi.

`print(name)` yuzi tanib olingan insonning ismini chop etadi.

`y1,x2,y2,x1 = faceLoc` bu kodda `y1,x2,y2,x1` o‘zgaruvchilariga `faceLoc`(kameradan olinayotgan rasmdan yuzni kordinatalarini qaytaradi) o‘zgaruvchisini yukladim.

`y1,x2,y2,x1 = y1*4,x2*4,y2*4,x1*4` bu kod y larning qiymatlarini to‘rt marta oshirib beradi.

`markAttendance()` funksiyasiga kameradan tanib olingan talaba ismini yo‘boramiz va funksiya talabaning ma’lumotlarini csv fayliga yozadi.

`cv2_imshow(imgS)` bu kod kameradan o‘qilayotgan ma’lumotni manitorda ko‘rsatib turadi.

`cv2.waitKey(0)` bunda klaviaturadan nolni bossam kamera ishlashdan to‘xtaydi.

Dasturning to‘liq kodi

```
import cv2

import numpy as np

import face_recognition

import os

from datetime import datetime

import glob

# Train datasetni yuklash

path = "/content/drive/MyDrive/Colab Notebooks/Face/tra
in"

images = []

className = []

myList = os.listdir(path)

print(myList)

# Fayldan name larni ajratib olish

for cl in myList:

  curImg = cv2.imread(f"{path}/{cl}")
```

```
  images.append(curImg)
  className.append(os.path.splitext(cl)[0])
print(className)
# csv faylni yangilash
with open("/content/drive/MyDrive/Colab Notebooks/Face/
information.csv","w+") as f:
  f.writelines('Name,Group,Course,Time')
#Train jarayoni Yakun
def findEncodings(image):
  encodeList = []
  for img in images:
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    encode = face_recognition.face_encodings(img)[0]
    encodeList.append(encode)
  return encodeList

def markAttendance(name):
  info = ["AB-225","4"]
  student = [
      "Saparov_Sunnat",
      "Juraqulov_Ibratjon",
      "Muminov_Baxtiyor",
      "Tojiyev_Sevdiyor",
      "Tuxtashev_Shohruh",
      "Ahmadjonov_Muhammadjon",
      "Ayapberginov_Raul",
      "Ibragimov_Dilshod",
      "Qarshiyev_Dilshod",
      "Urishbayev_Asadbek",
```

```
      "Ikromov_Aslbek",
      "Rahmonov_Hikmat",
      "Sherboyev_Ja'far",
      "Qurbonov_Ravshan",
      "Tursunov_Azim",
      "Shosayitov_Muhriddin",
      "Ozodov_Javohir"]
  summary = dict.fromkeys(student,info)
  with open("/content/drive/MyDrive/Colab Notebooks/Face/information.csv","r+") as f:
    nameList = []
    myDataList = f.readline()
    for line in myDataList:
      entry = line.split(',')
      nameList.append(entry[0])
    if name not in nameList:
      now = datetime.now()
      dtString = now.strftime('%H:%M:%S')
      f.writelines(f"\n{name},{','.join(summary[name])},{dtString}")


encodeListKnown = findEncodings(images)
print('Encoding Complete')


cap = glob.glob("/content/drive/MyDrive/Colab Notebooks/Face/test/*")


for i in range(len(cap)):
  img = cv2.imread(cap[i])
  imgS = cv2.resize(img, (0,0), None, 0.25, 0.25)
```

```
  facesCurFrame = face_recognition.face_locations(imgS)
  encodesCutFrame = face_recognition.face_encodings(img
S, facesCurFrame)


  for encodeFace, faceLoc in zip(encodesCutFrame, faces
CurFrame):
    matches = face_recognition.compare_faces(encodeList
Known,encodeFace)
    faceDis = face_recognition.face_distance(encodeList
Known,encodeFace)
    print(faceDis)
    matchIndex = np.argmin(faceDis)


    if matches[matchIndex]:
      name = className[matchIndex]
      print(name)
      y1,x2,y2,x1 = faceLoc
      y1,x2,y2,x1 = y1*4,x2*4,y2*4,x1*4
      cv2.rectangle(img, (x1,y1),(x2,y2),(0,255,0),2)
      #cv2.rectangle(img, (x2,y2-
35),(x2,y2),(0,255,0),cv2.FILLED)
      cv2.putText(imgS,name,(x1,y2),cv2.FONT_HERSHEY_CO
MPLEX,1,(255,255,255),2)
      markAttendance(name)
  cv2_imshow(imgS)
```