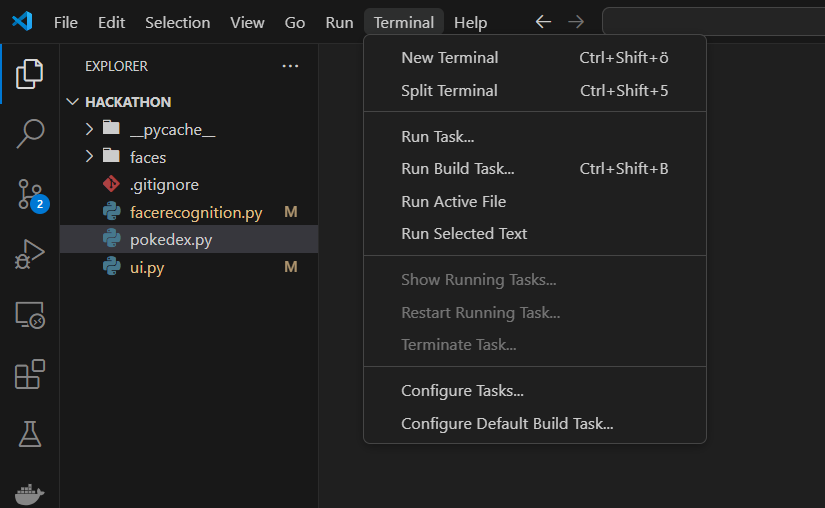
Pokedex Programmierungs-Guide

## Python Programm starten

**Visual Studio Code** öffnen

Erst muss ein neues Terminal geöffnet werden



Dort den Befehl **python pokedex.py** ausführen

## Programm auf Raspberry Pi ausführen

1. **face\_recognition.setup\_wincam()** ersetzen mit **face\_recognition.setup\_picam()**
2. in der Datei **facerecognition.py** in Zeile 8 das ***#*** löschen
3. Ganzes Projekt auf USB-Stick kopieren und im Raspberry Pi einfügen
4. Visual Studio auf dem Raspberry Pi starten
5. Schritte von oben wiederholen

## Basis aufbauen

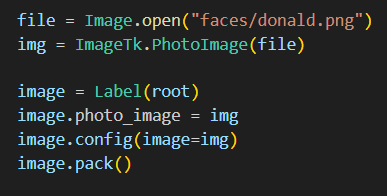
1. Eigene UI erstellen

Somit können beliebige Komponenten in das UI eingefügt werden. Das Label kann auch durch Buttons, Textfeldern, etc. ersetzt werden.



2. Bild einfügen

Statt **donald.png** wählt ihr eure selbst eingefügten Bilder aus. Hierfür könnt ihr auch mehrere Labels erstellen, und alle Bilder einfügen



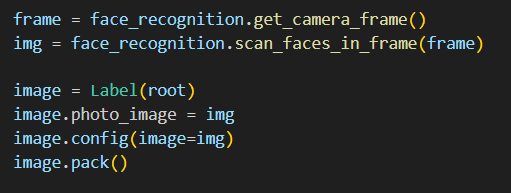
3. Bild von Kamera statt Datei laden

Die Funktion **get\_camera\_frame()** nimmt ein Bild aus der Kamera auf. Um dieses Bild in dem UI nun anzuzeigen, folgt ihr denselben Schritten aus Aufgabe 2 mit der neuen Variable **img.**



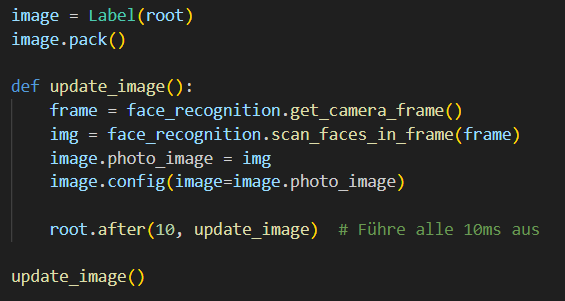
4. Gesichter erkennen

Von dem aus Aufgabe 3 geladenen Bild sollen nun eure Gesichter erkannt werden. Die Funktion **scan\_faces\_in\_frame()** nutzt KI, um alle Gesichter aus dem Bild zu erkennen. Alle Gesichter, die in dem Ordner **faces** als Bilddatei liegen, werden mit Namen angezeigt.



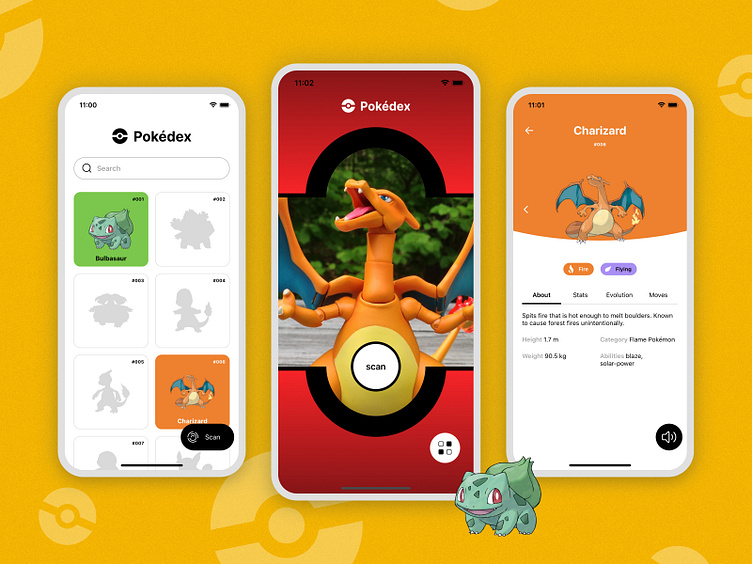
5. Live Video

Hiermit wird die Kamera live in dem UI angezeigt und durch die **scan\_faces\_in\_frame()** Funktion werden dauerhaft alle Gesichter erkannt



6. Gesamtes UI anpassen

Gestaltet das UI nach euren Wünschen. Nutzt hierfür Google oder ChatGPT, um eure Ideen umzusetzen. Lasst euer Programm wie ein echter Pokedex aussehen. Hier eine mögliche Inspiration:



Quelle: <https://dribbble.com/shots/22571703-Pok-dex-UI-Design-for-iOS-App>

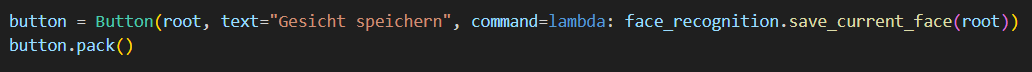
Beispiel für ChatGPT Anfrage: *Erstelle mir ein UI in Python und Tkinter, das aus einem Textfeld, einem Button und einem Hintergrundbild besteht.*

# Neue Gesichter speichern

Jetzt sollen unbekannte Gesichter, die bisher als „Unknown“ angezeigt werden, gespeichert und hinzugefügt werden können. Dafür kann ein Button hinzugefügt werden.

Folgt gerne zuerst der einfachen Lösung und probiert diese auf dem Raspberry Pi aus. Anschließend könnt ihr euch an der komplexeren Lösung probieren.

**Einfache Lösung**



In der Funktion **save\_current\_face** ist die Logik zum Speichern der Datei inkl. UI bereits implementiert, und kann durch Hinzfügen eines Buttons verwendet werden

**Komplexere Lösung**

Hierfür muss das Bild als Datei im Ordner **faces** abgespeichert werden. Dieses wird bei Neustart des Programms automatisch geladen. Ansonsten kann die **add\_face** Funktion aus der Datei **facerecognition** verwendet werden, um das Gesicht direkt zu laden.

## Liste von allen Gesichtern

Hier könnt ihr euch selbst etwas ausprobieren, und eine eigene UI erstellen :)

Es soll eine Liste erstellt werden, in der alle Gesichter angezeigt werden, die im **faces** Ordner liegen, sowie deren Namen. In der **facerecognition** Datei gibt es eine Funktion **get\_all\_faces**, die ein Array an allen Dateinamen enthält.