

```
← → Py_Chip_Eintrittssystem

Chip_EntrySystem.py × README.md

Chip_EntrySystem.py > EntrySystem > __init__
7 self.conn = sqlite3.connect(database_name,
8 self.create_tables()
9 self.cursor = self.conn.cursor()
10
11 def create_tables(self):
12     self.conn.execute('''
13         CREATE TABLE IF NOT EXISTS users (
14             chip_id INTEGER PRIMARY KEY,
15             username TEXT
16         )
17     ''')
18     self.conn.execute('')
```

PROBLEMS 9 OUTPUT TERMINAL PORTS COMMENTS

▼ **TERMINAL**

```
/usr/local/bin/python3 /Users/an/Desktop/Code/python3
on/Py_Chip_Eintrittssystem/Chip_EntrySystem.py
an@Anass-MacBook-Air Py_Chip_Eintrittssystem % /usr/local/bin/python3 /Users/an/Desktop/
thon/Py
_Chip_Eintrittssystem/Chip_EntrySystem.py
1. Register User
2. Check Access
3. Exit
Enter your choice:
```

Ln 7, Col 29 Spaces: 4 UTF-8 LF Python 3.12.0

Chip Eintrittssystem

1



Chip Eintrittssystem

2

```
Py_Chip_Eintrittssystem

Chip_EntrySystem.py M x README.md

Chip_EntrySystem.py > EntrySystem > get_current_time

40 self.conn.commit()
41 return f"Access granted for user: {username}"
42 return "Your Chip-Access is denied! Make sure you create an account"
43
44 def get_current_time(self):
45     return time.strftime('%Y-%m-%d %H:%M:%S')
46
47 def main():
48     database_name = 'entry_system.db'
49     entry_system = EntrySystem(database_name)
50
PROBLEMS 9 OUTPUT TERMINAL PORTS COMMENTS
▼ TERMINAL
y
an@Anass-MacBook-Air Py_Chip_Eintrittssystem % /usr/local/bin/python3 /Users/an/Desktop/Py_Chip_Eintrittssystem/Chip_EntrySystem.py
1. Register User
2. Check Access
3. Exit
Enter your choice: 2
Enter chip ID: 1
Your Chip-Access is denied! Make sure you create an account, if not, select 1
1. Register User
2. Check Access
3. Exit
Enter your choice: █

Ln 44, Col 32 Spaces: 4 UTF-8 LF Python 3.12.0
```

```
Py_Chip_Eintrittssystem

Chip_EntrySystem.py M x README.md

Chip_EntrySystem.py > EntrySystem > get_current_time

40         self.conn.commit()
41         return f"Access granted for user: {username}"
42         return "Your Chip-Access is denied! Make sure you create an account."
43
44     def get_current_time(self):
45         return time.strftime('%Y-%m-%d %H:%M:%S')
46
47     def main():
48         database_name = 'entry_system.db'
49         entry_system = EntrySystem(database_name)
50
PROBLEMS 9 OUTPUT TERMINAL PORTS COMMENTS
▼ TERMINAL
3. Exit
Enter your choice: 1
Enter username: Anas
User registered successfully. Your chip ID is 1
1. Register User
2. Check Access
3. Exit
Enter your choice: 2
Enter chip ID: 1
Access granted for user: Anas
1. Register User
2. Check Access
3. Exit
Enter your choice: █

0 9 0 Ln 44, Col 32 Spaces: 4 UTF-8 LF Python 3.12.0
```

Chip Eintrittssystem

3

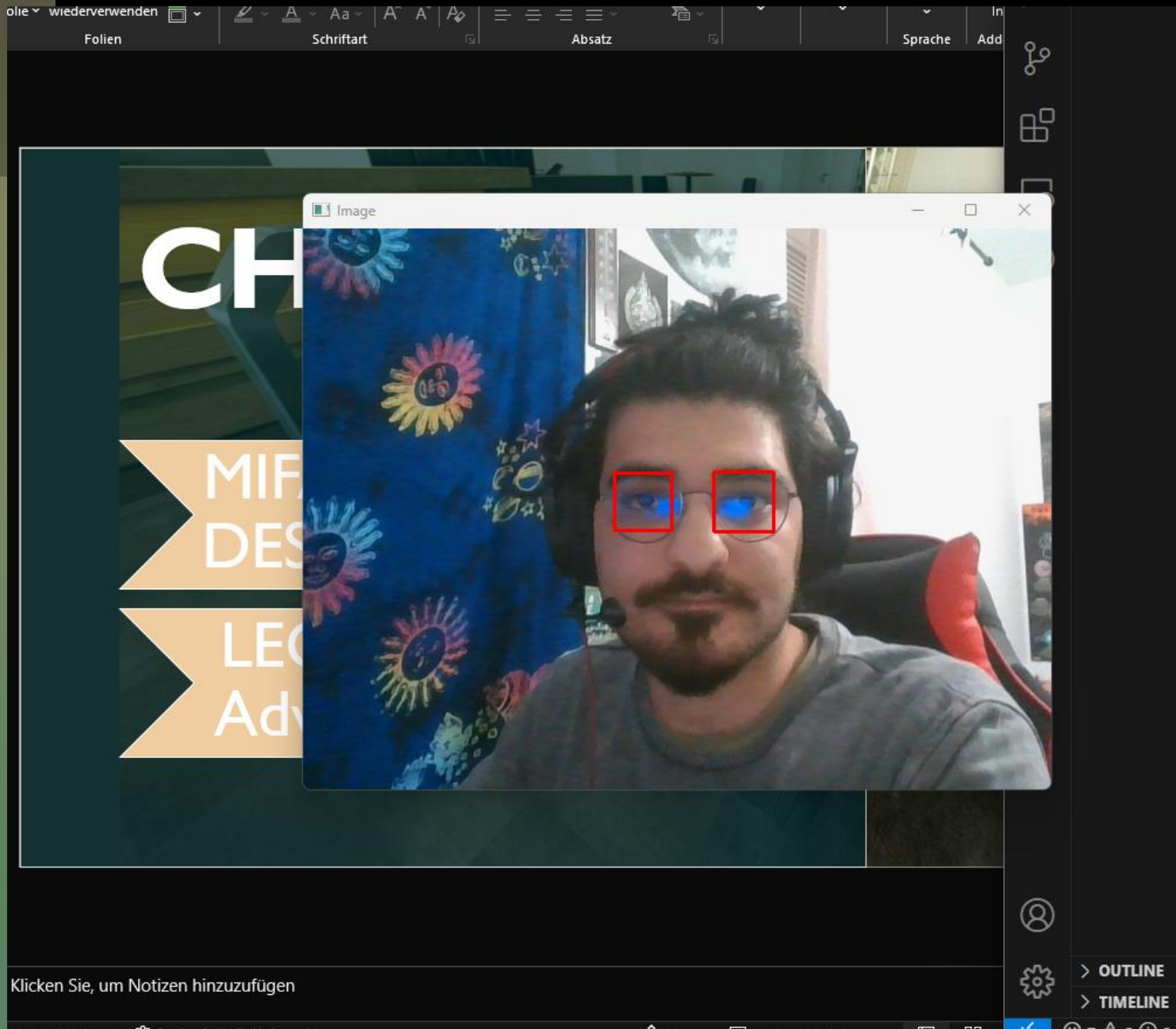


SCHEUNENTOR
Sichere Zutrittssysteme

Iris Detection

1

```
1 import time
2 import cv2
3
4 # Verwende die integrierte Haar Cascade XML-Datei für die Erkennung von Augen.
5 eyeCascadeClassifier = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_eye.xml')
6
7 def detect_objects(image, objectClassifier):
8     gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
9     objects = objectClassifier.detectMultiScale(gray_image, scaleFactor=1.3, minNeighbors=5, minSize=(30, 30))
10    return objects
11
12 def draw(photo):
13     image = photo.copy()
14     eyes = detect_objects(image, eyeCascadeClassifier)
15
16     # Zeichne Rechtecke um beide erkannten Augen.
17     for (x, y, w, h) in eyes:
18         cv2.rectangle(image, (x, y), (x + w, y + h), (0, 0, 255), 2)
19
20     cv2.imshow('Image', image)
21     key = cv2.waitKey(10)
22
23     return key
24
25 def main():
26     camera = cv2.VideoCapture(0)
27     while True:
28         success, photo = camera.read()
29         key = draw(photo)
30         if key > 0:
31             break
32
33 if __name__ == '__main__':
34     main()
```



Iris Detection

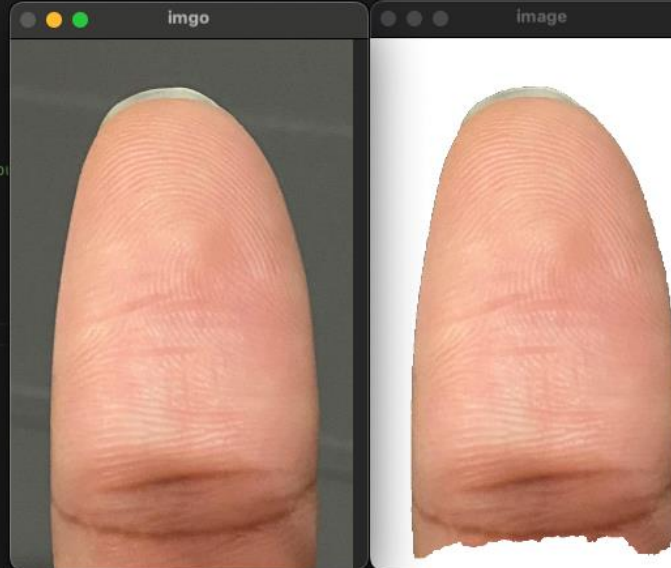
2

```
Fingerprint-Matching

remove_back.py x preprocess.py sift_detect.py

remove_back.py > ...
1 import cv2
2 import numpy as np
3
4
5 # It reads an image from a file called 'f1.jpeg'
6 # using cv2.imread and displays it in a window using cv2.imshow.
7 #imgo = cv2.imread('./input/f2.jpg')
8 imgo = cv2.imread('./input/f1.jpeg')
9 cv2.imshow("imgo",imgo)
10
11
12 #Removing the background
13 height, width = imgo.shape[:2] #The code then proceeds to remove the backgro
14
15 #Create a mask holder
16 mask = np.zeros(imgo.shape[:2],np.uint8)
17
18 #Grab Cut the object
19 bgdModel = np.zeros((1,65),np.float64)
20 fgdModel = np.zeros((1,65),np.float64)
21
22 #Hard Coding the Rect.. The object must lie within this rect.
23 rect = (10,10,width-30,height-30)
24 cv2.grabCut(imgo,mask,rect,bgdModel,fgdModel,5,cv2.GC_INIT_WITH_RECT)
25 mask = np.where((mask==2)|(mask==0),0,1).astype("uint8")
26 img1 = imgo*mask[:, :, np.newaxis]
27
28 #
29 #Get the background
30 background = cv2.absdiff(imgo,img1)
31
32 #Change all pixels in the background that are not black to white
33 background[np.where((background > [0,0,0]).all(axis = 2)) ] = [255,255,255]
34
35 #Add the background and the image
36 final = background + img1
37
PROBLEMS OUTPUT TERMINAL PORTS COMMENTS
▼ TERMINAL
/Users/an/miniconda3/bin/python /Users/an/Desktop/Code/python/Fingerprint-Matching/remove_back.py
o (base) an@Anass-Mac-mini Fingerprint-Matching % /Users/an/miniconda3/bin/python /Users/an/Desktop/Code/python/Fingerprint-Matching/remove_back.py

Ln 20, Col 39 Spaces: 4 UTF-8 LF Python 3.11.4 ('base': conda) Go Live
```



Fingerprint-Matching-Programm


1

Fingerprint-Matching

remove_back.py M preprocess.py M x sift_detect.py •

```
preprocess.py > plot_images
1 import cv2
2 import numpy as np
3 import matplotlib.pyplot as plt
4 from skimage.feature import hessian_matrix, hessian_matrix_eigvals
5
6 def detect_ridges(gray, sigma=3.0):
7     #This function uses the Hessian matrix to detect edges (ridges)
8     H_elems = hessian_matrix(gray, sigma=sigma, order='rc')
9     maxima_ridges, minima_ridges = hessian_matrix_eigvals(H_elems)
10    return maxima_ridges, minima_ridges
11
12 def plot_images(*images):
13     #This function is used to display and save images.
14     #It creates a figure with the specified images and saves it as a png file.
15     images = list(images)
16     n = len(images)
17     fig, ax = plt.subplots(ncols=n, sharey=True)
18     for i, img in enumerate(images):
19         ax[i].imshow(img, cmap='gray')
20         ax[i].axis('off')
21         extent = ax[i].get_window_extent().transformed(fig.dpi_scale_trans.inverted())
22         plt.savefig('fig'+str(i)+'.png', bbox_inches=extent)
23     plt.subplots_adjust(left=0.03, bottom=0.03, right=0.97, top=0.97)
24     plt.show()
25
26
27
28
29 def main(): #This is the main function of the script, where the script starts
30
31     # ----- Step 1: import the image whose background has been removed
32     # Step 1: The input image is loaded using cv2.imread.
33     # This image should already have its background removed, a white background.
34
35     img = cv2.imread("input.jpg",1)
36
37
```

Figure 1



x=185. y=20. [0.0000]

PROBLEMS OUTPUT TERMINAL PORTS COMMENTS

TERMINAL

```
/Users/an/miniconda3/bin/python /Users/an/Desktop/Code/python/Fingerprint-Matching/preprocess.py
(base) an@Anass-Mac-mini Fingerprint-Matching % /Users/an/miniconda3/bin/python /Users/an/Desktop/Code/python/Fingerprint-Matching/preprocess.py
/Users/an/Desktop/Code/python/Fingerprint-Matching/preprocess.py:8: FutureWarning: use_gaussian_derivatives currently defaults to False, but will change to True in a future version. Please specify this argument explicitly to maintain the current behavior
  H_elems = hessian_matrix(gray, sigma=sigma, order='rc')
```

Ln 14, Col 80 Tab Size: 4 UTF-8 LF Python 3.11.4 ('base': conda) Go Live

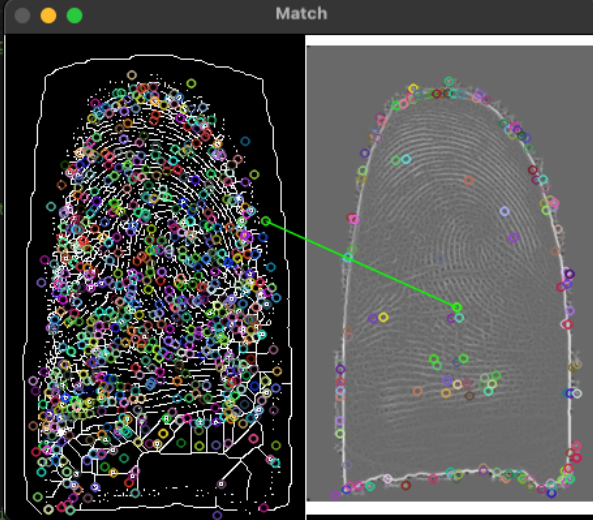
Fingerprint-Matching-Programm

2

```

Fingerprint-Matching
remove_back.py M preprocess.py M sift_detect.py X
output > sift_detect.py > ...
1 import numpy as np
2 import cv2 as cv
3 import glob, os
4
5
6 # It sets a minimum match count to 15,
7 # which means that at least 15 matching features are required to consider two fingerprint images as a match.
8 MIN_MATCH_COUNT = 15
9
10
11
12 # PHOTO TO FIND FEATURE POINTS
13 # It loads a reference fingerprint image (specified as 'input_img') and converts it to grayscale
14
15 #input_img = cv.imread('./Fingerprint-Matching/fig0.png')
16 input_img = cv.imread('./output/1.png')
17 input_img = input_img.astype('uint8')
18 gray = cv.cvtColor(input_img, cv.COLOR_BGR2GRAY)
19 sift = cv.SIFT_create() # Use cv.SIFT_create() instead of cv.xfeatures2d.SIFT_create()
20 # uses the Scale-Invariant Feature Transform (SIFT) algorithm to detect key points (feature points)
21 kp = sift.detect(input_img, None)
22 img1 = cv.drawKeypoints(input_img, kp, input_img)
23
24 flag = 0
25
26 os.chdir("./")
27 for file in glob.glob("*.png"):
28
29     frame = cv.imread(file) # It reads the image file using cv.imread and assigns it to 'frame'
30     frame = frame.astype('uint8')
31     gray1 = cv.cvtColor(frame, cv.COLOR_BGR2GRAY)
32
33     # It uses the SIFT (Scale-Invariant Feature Transform) algorithm for feature detection
34     sift = cv.SIFT_create() # Use cv.SIFT_create() instead of cv.xfeatures2d.SIFT_create()
35
36
37     kp = sift.detect(frame, None)

```



PROBLEMS OUTPUT **TERMINAL** PORTS COMMENTS

```

/Users/an/miniconda3/bin/python /Users/an/Desktop/Code/python/Fingerprint-Matching/output/sift_detect.py
(base) an@Anass-Mac-mini Fingerprint-Matching % /Users/an/miniconda3/bin/python /Users/an/Desktop/Code/python/Fingerprint-Matching/output/sift_detect.py

```

Ln 12, Col 31 Spaces: 4 UTF-8 LF Python 3.11.4 ('base': conda) Go Live

Fingerprint-Matching-Programm

3


```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6    <meta http-equiv="X-UA-Compatible" content="ie=edge">
7    <title>SCHEUNENTOR</title>
8    <script defer src="face-api.min.js"></script>
9    <script defer src="script.js"></script>
10   <style>
11     body {
12       margin: 0;
13       padding: 0;
14       width: 100vw;
15       height: 100vh;
16       display: flex;
17       justify-content: center;
18       align-items: center;
19     }
20
21     canvas {
22       position: absolute;
23     }
24   </style>
25 </head>
26 <body>
27   <video id="video" width="720" height="560" autoplay muted></video>
28 </body>
29 </html>
```

Java Script Face Detection

1

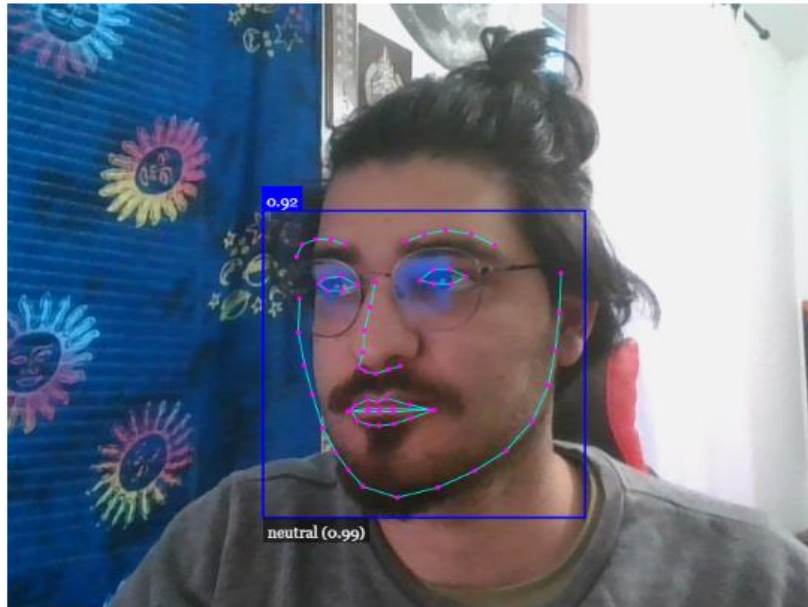


SCHEUNENTOR
Sichere Zutrittsysteme

```
1  const video = document.getElementById('video')
2
3  Promise.all([
4    faceapi.nets.tinyFaceDetector.loadFromUri('/models'),
5    faceapi.nets.faceLandmark68Net.loadFromUri('/models'),
6    faceapi.nets.faceRecognitionNet.loadFromUri('/models'),
7    faceapi.nets.faceExpressionNet.loadFromUri('/models')
8  ]).then(startVideo)
9
10 function startVideo() {
11   navigator.getUserMedia(
12     { video: {} },
13     stream => video.srcObject = stream,
14     err => console.error(err)
15   )
16 }
17
18 video.addEventListener('play', () => {
19   const canvas = faceapi.createCanvasFromMedia(video)
20   document.body.append(canvas)
21   const displaySize = { width: video.width, height: video.height }
22   faceapi.matchDimensions(canvas, displaySize)
23   setInterval(async () => {
24     const detections = await faceapi.detectAllFaces(video, new faceapi.TinyFaceDetectorOptions()).with
25     const resizedDetections = faceapi.resizeResults(detections, displaySize)
26     canvas.getContext('2d').clearRect(0, 0, canvas.width, canvas.height)
27     faceapi.draw.drawDetections(canvas, resizedDetections)
28     faceapi.draw.drawFaceLandmarks(canvas, resizedDetections)
29     faceapi.draw.drawFaceExpressions(canvas, resizedDetections)
30   }, 100)
31 })
```

Java Script Face Detection

2



Java Script Face Detection

3

Den Quellcode finden Sie auf Github:

<https://github.com/anasm20>

Py_Iris-Detection

https://github.com/anasm20/Py_Iris-Detection

JS_Face-Detection

https://github.com/anasm20/JS_Face-Detection

Py_Chip_Eintrittssystem

https://github.com/anasm20/Py_Chip_Eintrittssystem

Py_Fingerprint-Matching

<https://github.com/anasm20/Fingerprint-Matching>