```
3  # import the necessary packages
4  from imutils.video import VideoStream
5  from imutils.video import FPS
6  import face_recognition
7  import imutils
8  import pickle
9  import time
10 import cv2
11
12 #Initialize 'currentname' to trigger only when a new person is identified.
13 currentname = "unknown"
14 #Determine faces from encodings.pickle file model created from train_model.py
15 encodingsP = "encodings.pickle"
16
17 # load the known faces and embeddings along with OpenCV's Haar
18 # cascade for face detection
19 print("[INFO] loading encodings + face detector...")
20 data = pickle.loads(open(encodingsP, "rb").read())
21
22 # initialize the video stream and allow the camera sensor to warm up
23 # Set the ser to the followng
24 # src = 0 : for the build in single web cam, could be your laptop webcam
25 # src = 2 : I had to set it to 2 inorder to use the USB webcam attached to my laptop
26 vs = VideoStream(src=2,framerate=10).start()
27 #vs = VideoStream(usePiCamera=True).start()
28 time.sleep(2.0)
29
30 # start the FPS counter
31 fps = FPS().start()
32
33 # loop over frames from the video file stream
34 while True:
35         # grab the frame from the threaded video stream and resize it
36         # to 500px (to speedup processing)
```

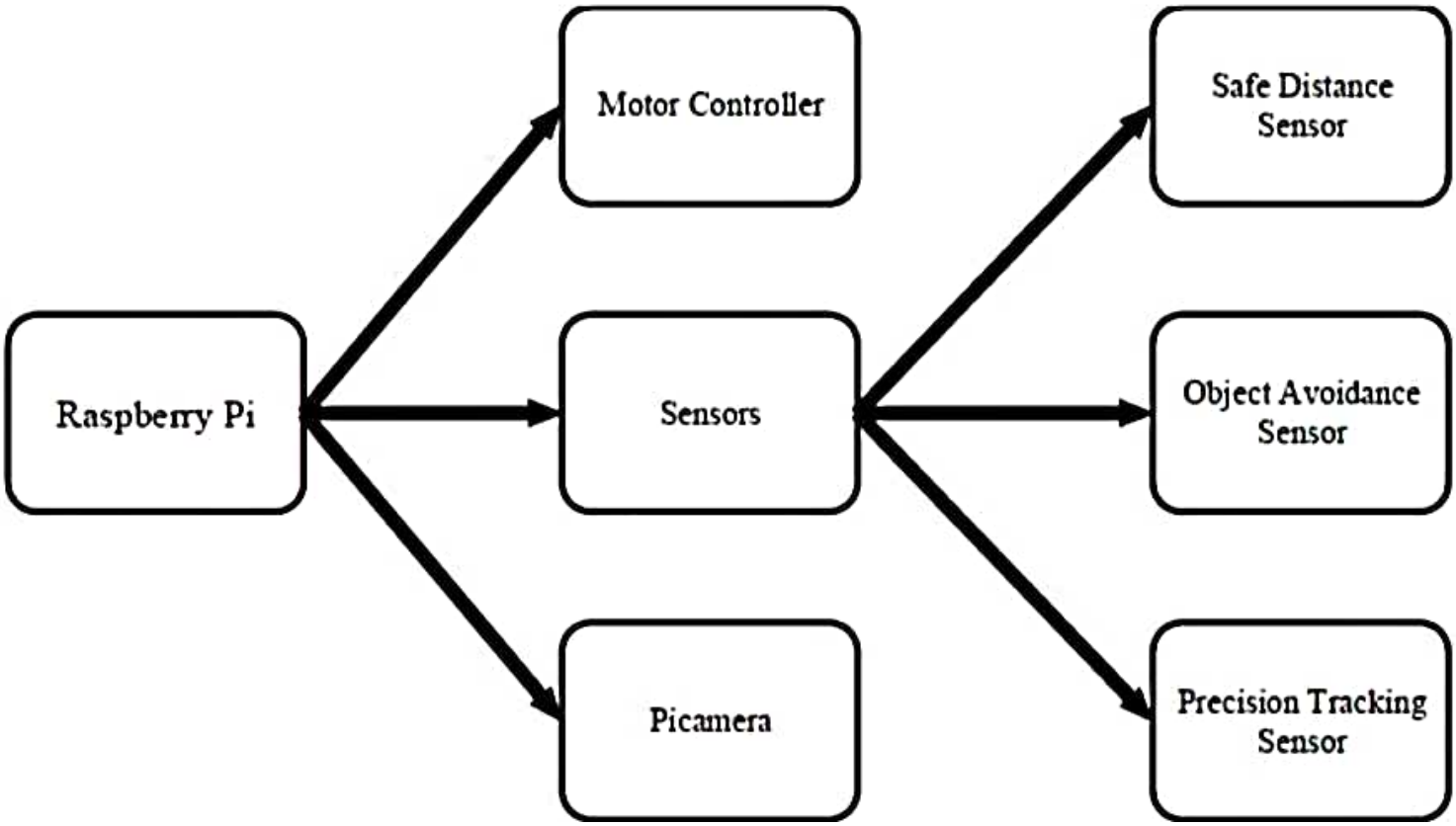
```
frame = imutils.resize(frame, width=500)
# Detect the fce boxes
boxes = face_recognition.face_locations(frame)
# compute the facial embeddings for each face bounding box
encodings = face_recognition.face_encodings(frame, boxes)
names = []

# loop over the facial embeddings
for encoding in encodings:
        # attempt to match each face in the input image to our known
        # encodings
        matches = face_recognition.compare_faces(data["encodings"],
                encoding)
        name = "Unknown" #if face is not recognized, then print Unknown

        # check to see if we have found a match
        if True in matches:
                # find the indexes of all matched faces then initialize a
                # dictionary to count the total number of times each face
                # was matched
                matchedIdxs = [i for (i, b) in enumerate(matches) if b]
                counts = {}

                # loop over the matched indexes and maintain a count for
                # each recognized face face
                for i in matchedIdxs:
                        name = data["names"][i]
                        counts[name] = counts.get(name, 0) + 1

                # determine the recognized face with the largest number
                # of votes (note: in the event of an unlikely tie Python
                # will select first entry in the dictionary)
                name = max(counts, key=counts.get)
```


Raspberry Pi
Motor Controller
Sensors
Picamera
Safe Distance Sensor
Object Avoidance Sensor
Precision Tracking Sensor