Progettazione e produzione multimediale

Sistema di riconoscimento e raccolta di testo ed emozioni

Leonardo Gori
Cosimo Michelagnoli

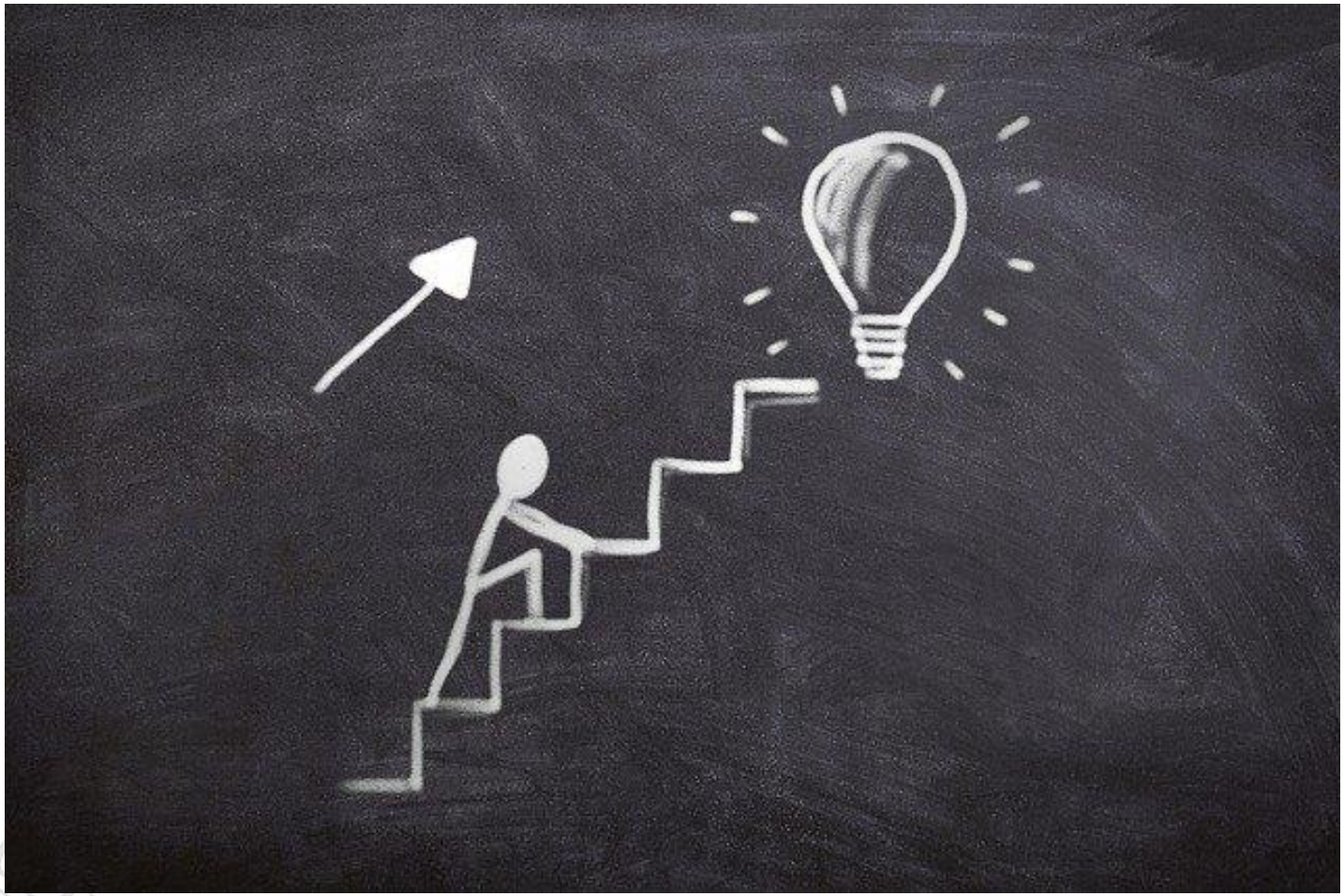
14/09/2020

L'idea…

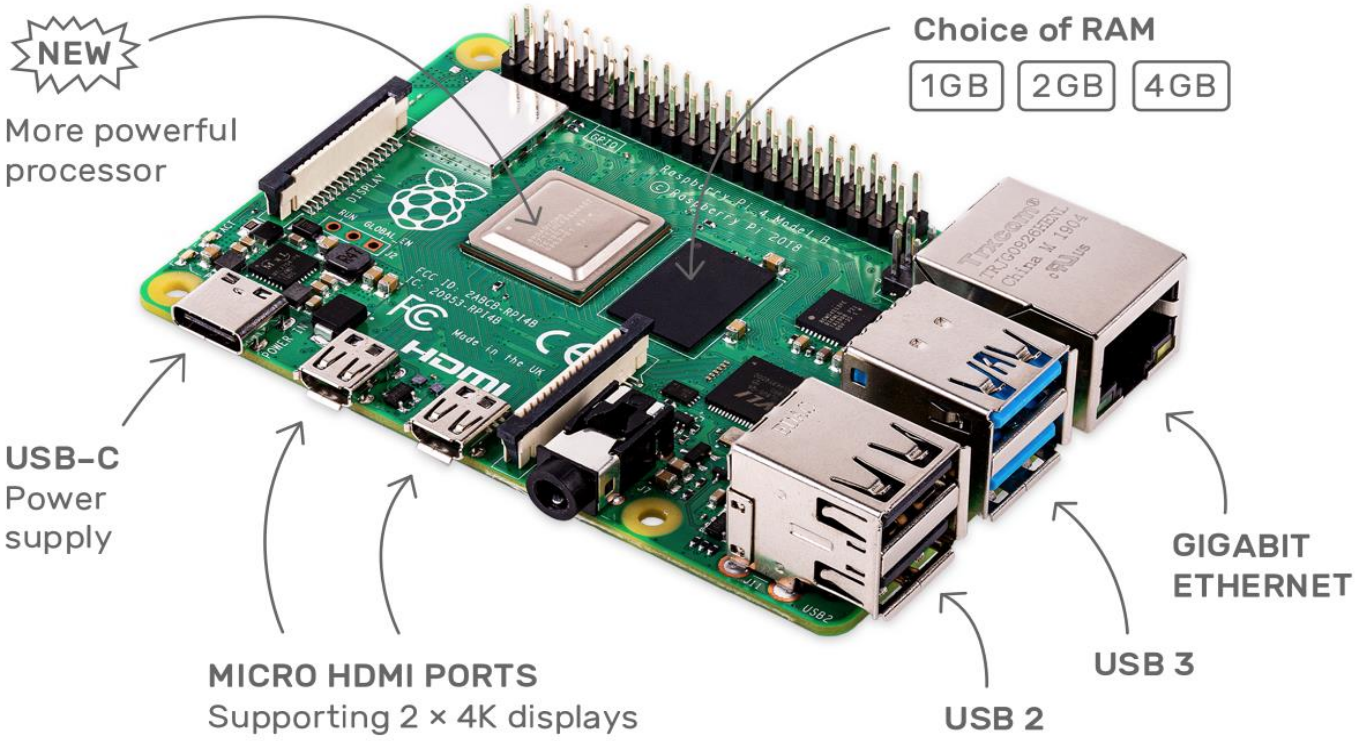
Aspetti operativi del progetto:

- Fattibilità economica del progetto: *(mappatura del flusso del valore)*
- Fattibilità in termini di dimensioni fisiche: *(è utilizzabile in un match?)*
- Fattibilità in termini di competenze progettuali: (cosa possiamo fare?)

L'idea per diventare progetto deve essere realizzabile…



Raspberry pi 4

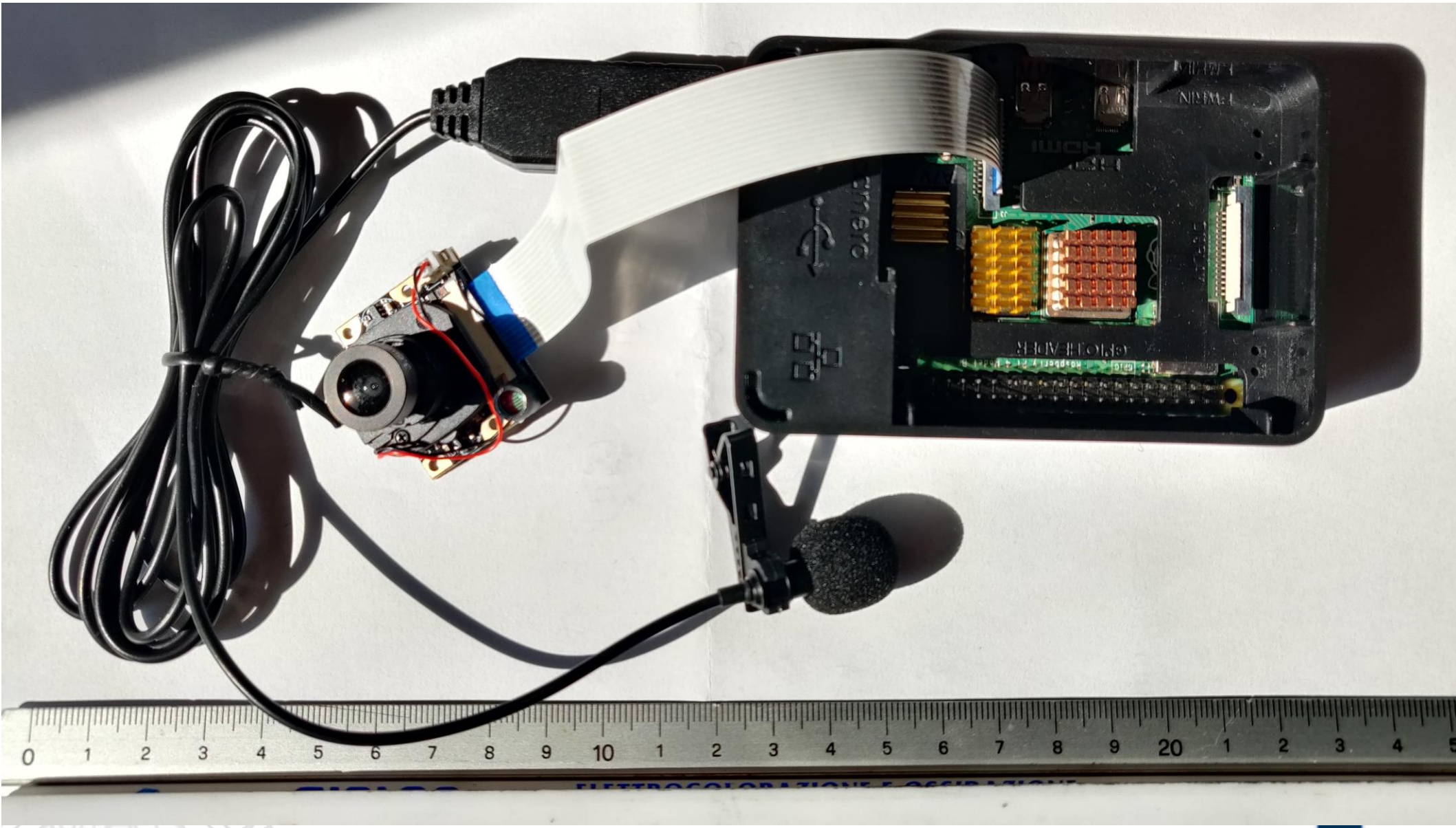


Periferiche I/O

Camera

Microfono

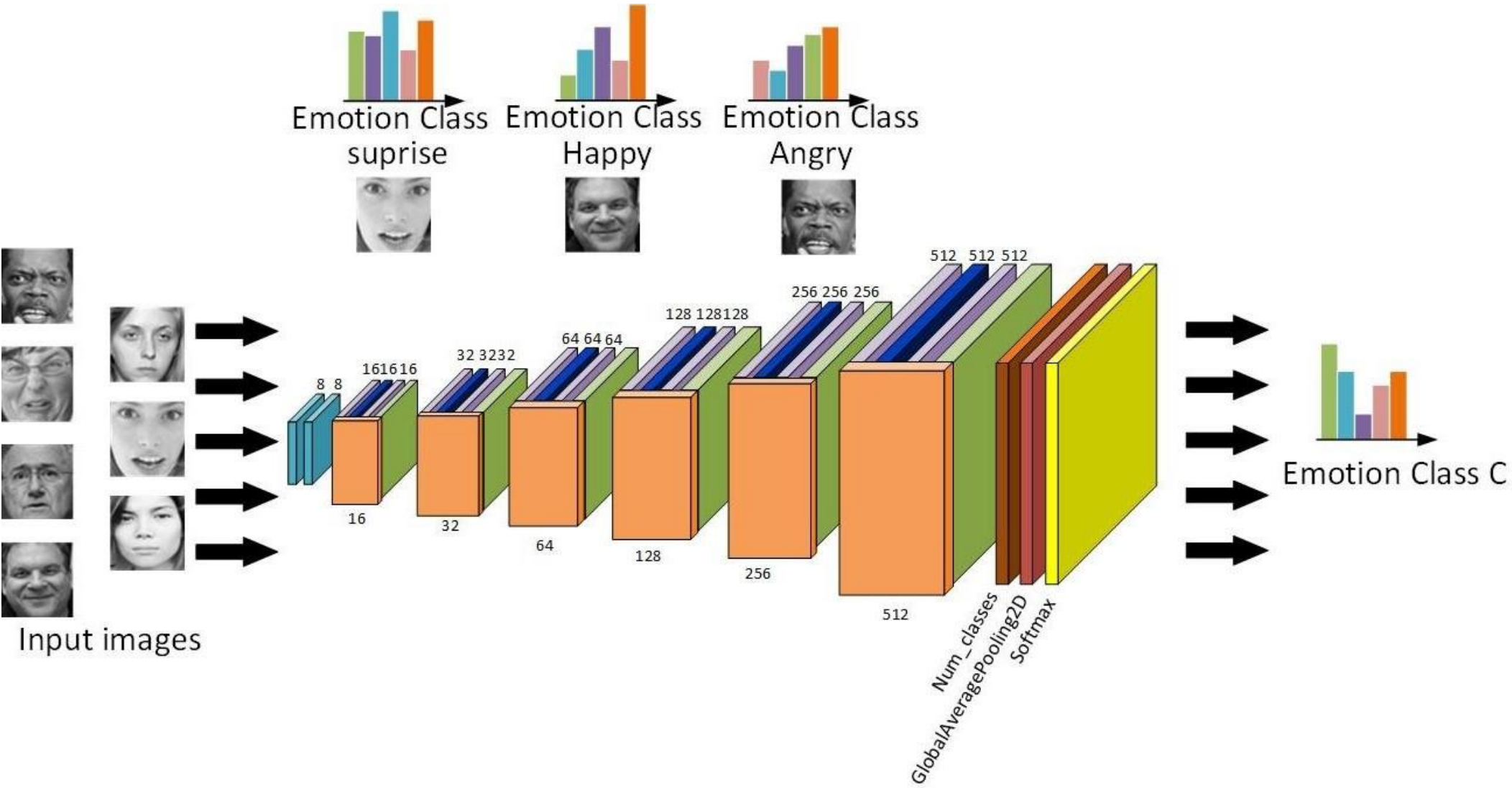
Dimensioni



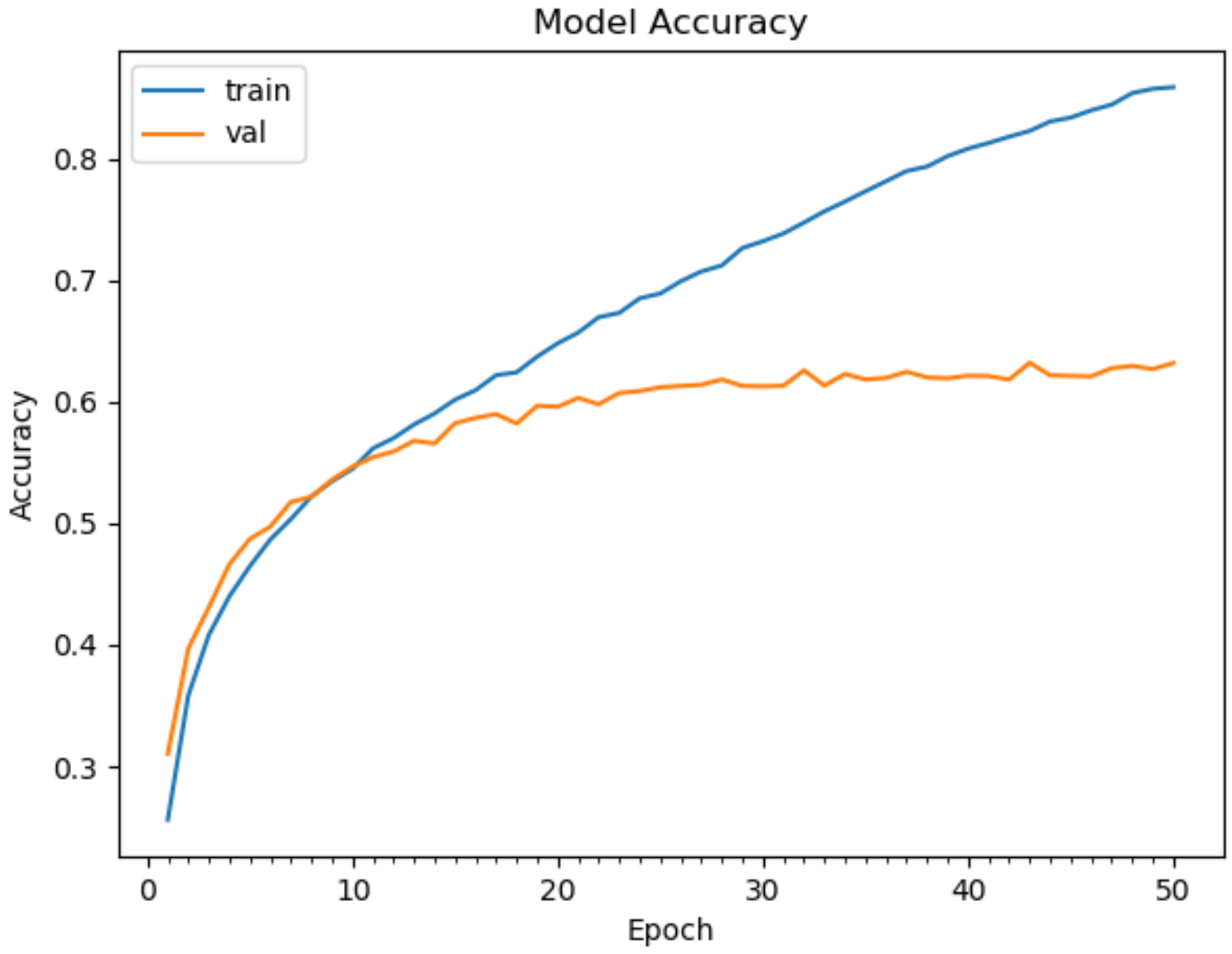
Software utilizzato:

- Emotion detection using deep learning
- Google Speech Recognition API

Emotion detection



Accuratezza



```
class Emotions(Thread):

    def __init__(self):
        super().__init__()
        self.vectEmotion = []
        os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'

        # Create the model
        self.model = Sequential()

        self.model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(48, 48, 1)))
        self.model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
        self.model.add(MaxPooling2D(pool_size=(2, 2)))
        self.model.add(Dropout(0.25))

        self.model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
        self.model.add(MaxPooling2D(pool_size=(2, 2)))
        self.model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
        self.model.add(MaxPooling2D(pool_size=(2, 2)))
        self.model.add(Dropout(0.25))

        self.model.add(Flatten())
        self.model.add(Dense(1024, activation='relu'))
        self.model.add(Dropout(0.5))
        self.model.add(Dense(7, activation='softmax'))

        # emotions will be displayed on your face from the webcam feed

        self.model.load_weights('model.h5')

        # prevents openCL usage and unnecessary logging messages
        cv2.ocl.setUseOpenCL(False)
        # dictionary which assigns each label an emotion (alphabetical order)
        self.emotion_dict = {0: "Arrabbiato", 1: "Disgustato", 2: "Impaurito", 3: "Felice", 4: "Neutrale", 5: "Triste",
                             6: "Sorpreso"}

        # start the webcam feed
        self.cap = cv2.VideoCapture(0)
```

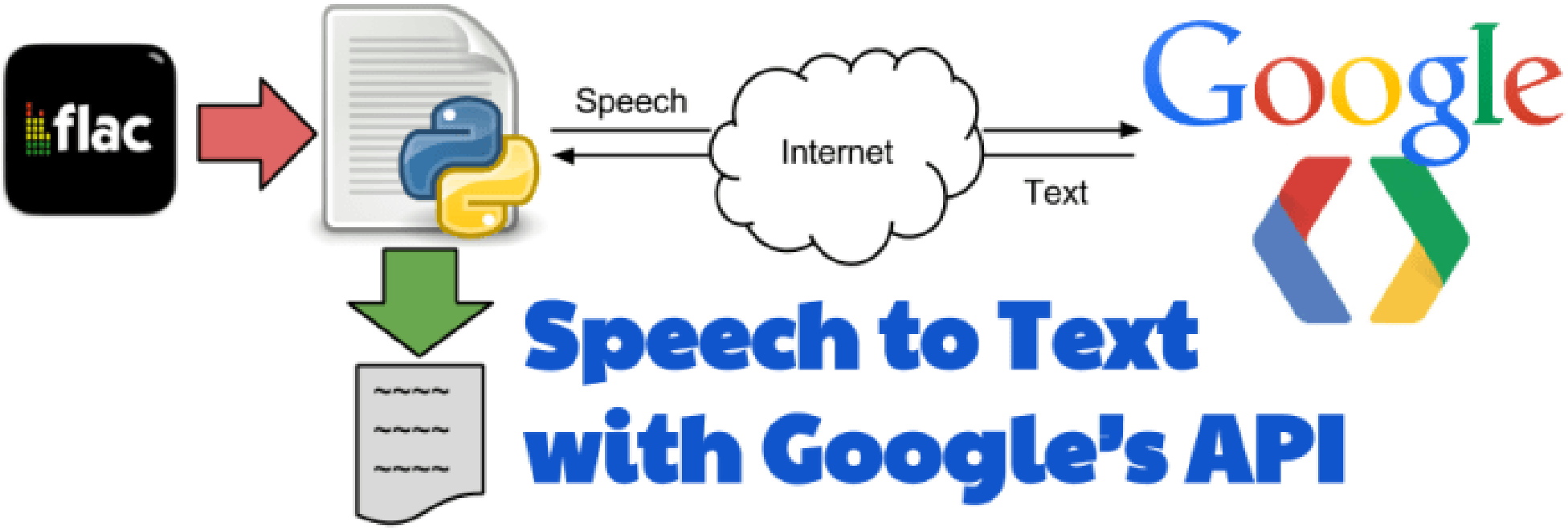
```
def run(self):

    while True:

        # Find haar cascade to draw bounding box around face
        ret, frame = self.cap.read()
        if not ret:
            break
        facecasc = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        color = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        faces = facecasc.detectMultiScale(gray, scaleFactor=1.3, minNeighbors=5)

        for (x, y, w, h) in faces:
            cv2.rectangle(frame, (x, y - 50), (x + w, y + h + 10), (255, 0, 0), 2)
            roi_gray = gray[y:y + h, x:x + w]
            roi_color = color[y:y + h, x:x + w]
            self.cropped_img = np.expand_dims(np.expand_dims(cv2.resize(roi_gray, (48, 48)), -1), 0)
            self.image = np.expand_dims(np.expand_dims(cv2.resize(roi_color, (305, 305)), -1), 0)
            prediction = self.model.predict(self.cropped_img)
            maxindex = int(np.argmax(prediction))
            self.vectEmotion.append(maxindex)
            cv2.putText(frame, self.emotion_dict[maxindex], (x + 20, y - 60), cv2.FONT_HERSHEY_SIMPLEX, 1,
                        (255, 255, 255),
                        2, cv2.LINE_AA)

        cv2.imshow('Video', cv2.resize(frame, (1600, 960), interpolation=cv2.INTER_CUBIC))
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    self.cap.release()
    cv2.destroyAllWindows()

def getEmotion(self):
    return self.emotion_dict[self.most_common(self.vectEmotion)]

def getCropeImage(self):
    return self.image

def setReady(self):
    self.vectEmotion.clear()
```

1

2

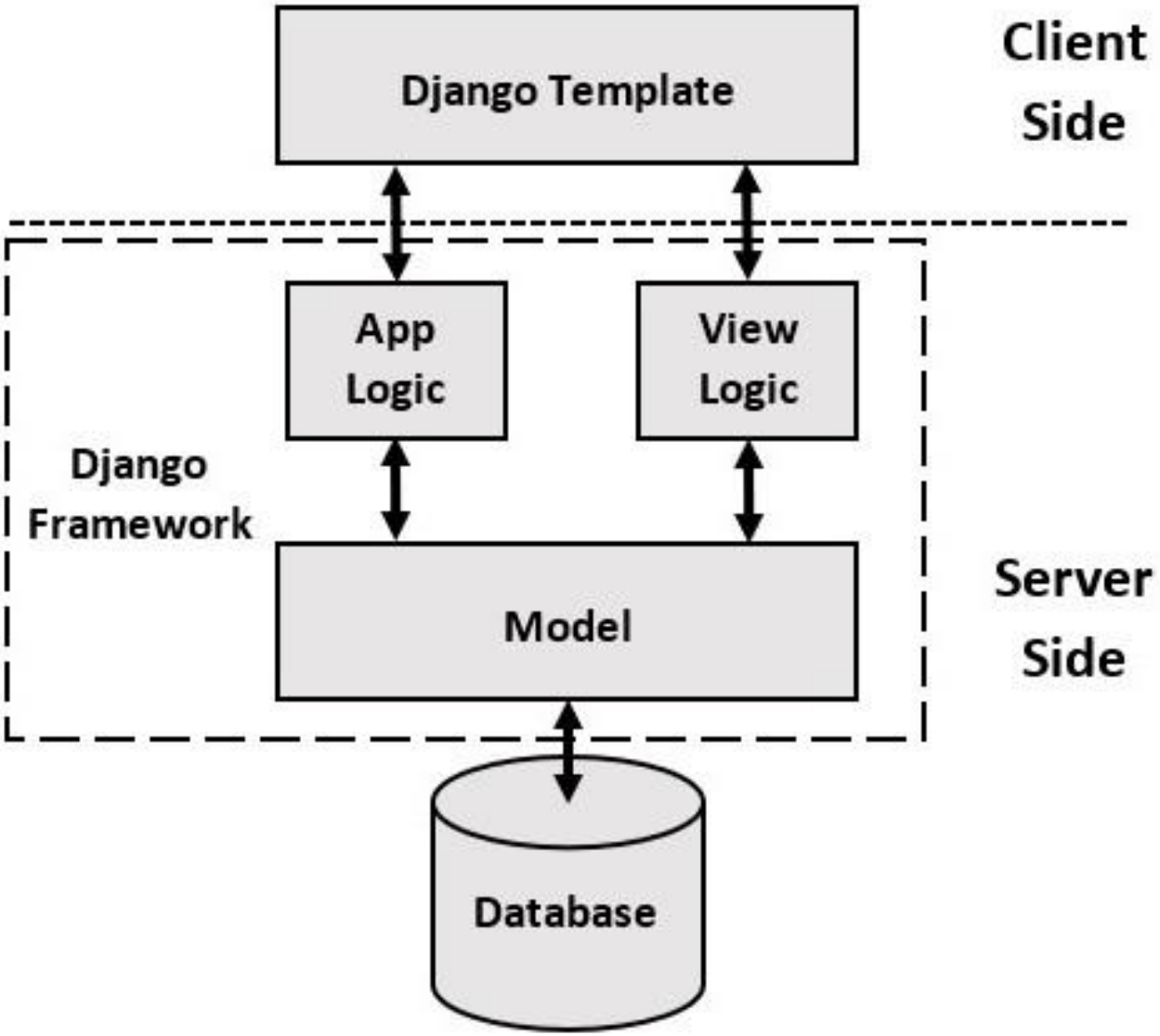
3

Speech Recognition

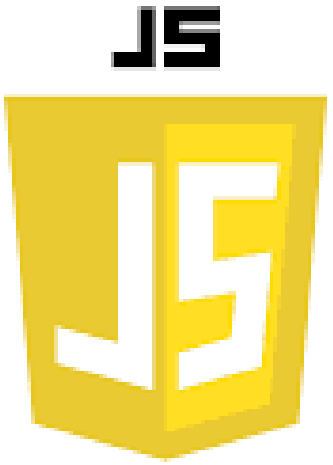


```
class SpeechRecognition(Thread):
    def __init__(self, emotion):
        super().__init__()
        self.emotion = emotion

    def run(self):
        # obtain audio from the microphone

        while True:
            r = sr.Recognizer()
            with sr.Microphone() as source:
                print("Say something!")
                self.emotion.setReady()
                audio = r.listen(source)

            try:
                # for testing purposes, we're just using the default API key
                # to use another API key, use `r.recognize_google(audio, key="GOOGLE_SPEECH_RECOGNITION_API_KEY")`
                # instead of `r.recognize_google(audio)`
                text = r.recognize_google(audio, language="it-IT")
                IMIR = self.emotion.getCropeImage().reshape(305, 305, 3)
                img = Image.fromarray(IMIR)
                img.save('prova.png')
                with open("prova.png", "rb") as file:
                    img = base64.b64encode(file.read())
                # img = Image.open(io.BytesIO(base64.b64decode(img)))
                print("Google Speech Recognition thinks you said: " + text
                      + " while your mood was " + self.emotion.getEmotion())
                str_img = str(img)
                str_img = str_img[2:len(str_img) - 1]
                today = date.today()
                my_date = today.strftime("%Y-%m-%d")
                t = Table(date=my_date, speech_text=text, emotion=self.emotion.getEmotion(), image=str_img)
                t.save()

            except sr.UnknownValueError:
                print("Google Speech Recognition could not understand audio")
            except sr.RequestError as e:
                print("Could not request results from Google Speech Recognition service; {0}".format(e))
```

Django

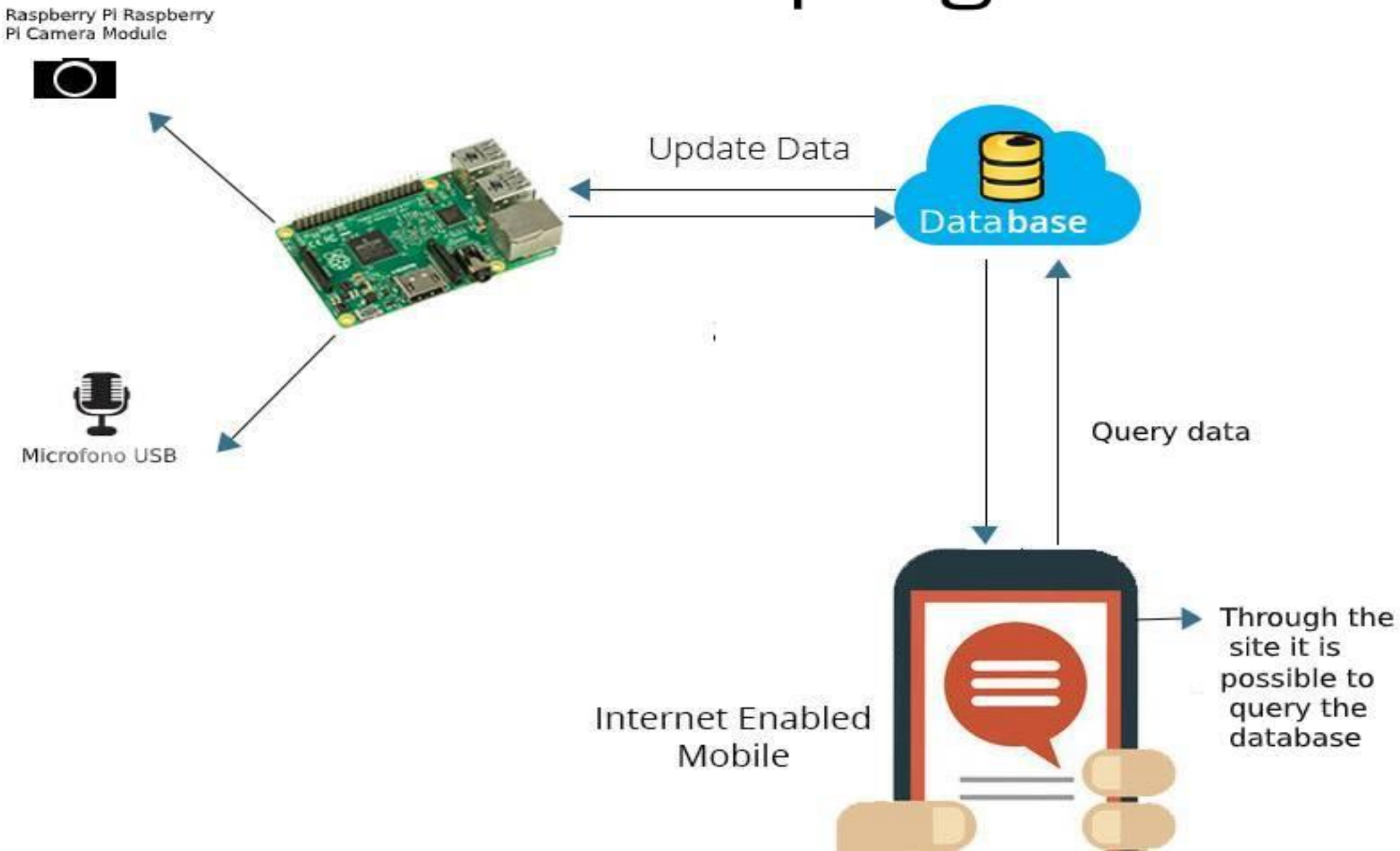

Django Template
Client Side
App Logic
View Logic
Django Framework
Model
Server Side
Database

Interfaccia grafica

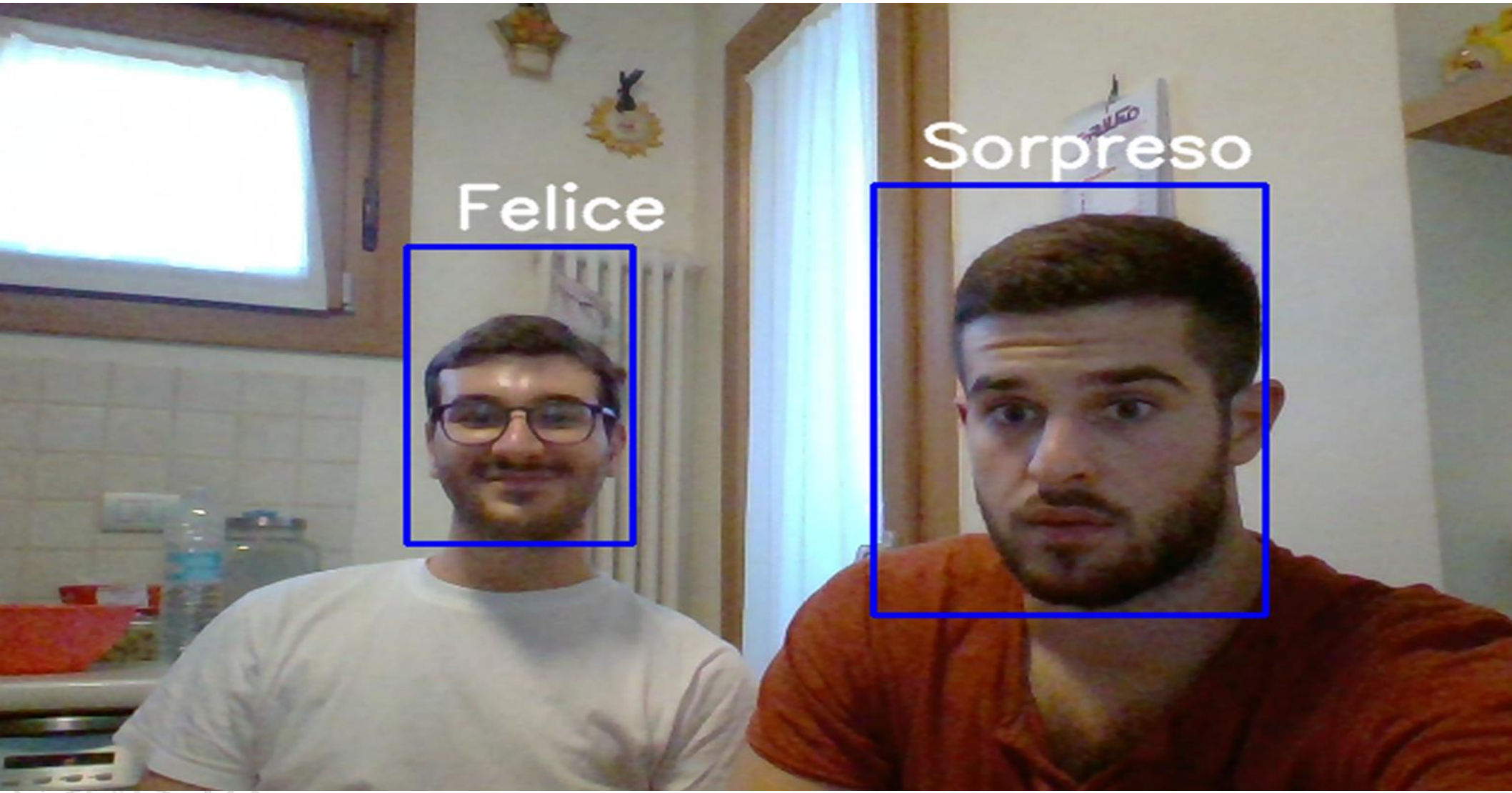


Bootstrap

Architettura progetto



Risultati ottenuti



- Fattibilità economica del progetto

- Fattibilità in termini di dimensioni fisiche

- Fattibilità in termini di competenze progettuali

Sitografia:

Link Riconoscimento delle emozioni:
https://fablab.ruc.dk/facial-expression-recognition-on-a-raspberry-pi/
https://github.com/atulapra/Emotion-detection

Link Riconoscimento del testo:
https://github.com/Uberi/speech_recognition