

Identity Authentication for Online Exams Using Face Recognition and Liveness Detection

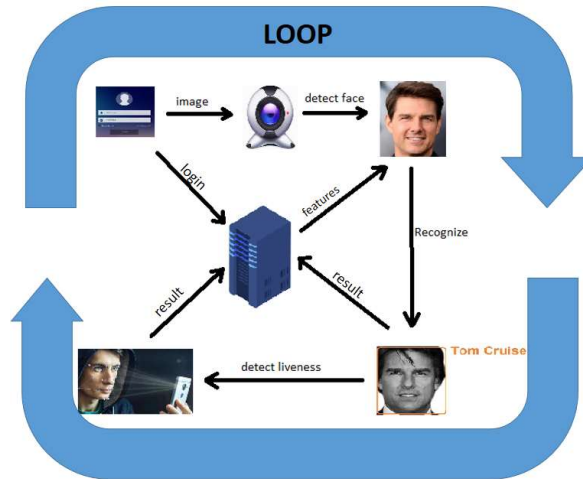
1.1. ABSTRACT

Our objective is to prevent cheating in online exams. This phenomenon can't be done with password login only, so face recognition and its anti-spoofing mechanism "liveness detection" are added to have anti-cheating system where it recognizes and detects the presence of the students, then returns the results. This system runs in the client side via the web browser because face recognition on server side for a large many students will be slow and hard to execute, so the server can handle data, and based on the ID of the logged-in user, it sends an embedding vector representing his/her face to the client web browser to be used in the face recognition process. Both face recognition and liveness detection use deep learning models, but that for liveness detection, which was built and trained in python and then converted to Tensorflow.JS model.

1.2. SYSTEM ARCHITECTURE

As presented in below figure first the system takes frames using webcam to detect the face and crop, then the server sends the features to recognize it. The features sent by the server correspond to the face of logged student. The server is notified about the output of face recognition and liveness detection algorithms: whether the face corresponds to the considered student, and whether the captured images are live or not. And this process repeats throughout the exam time.

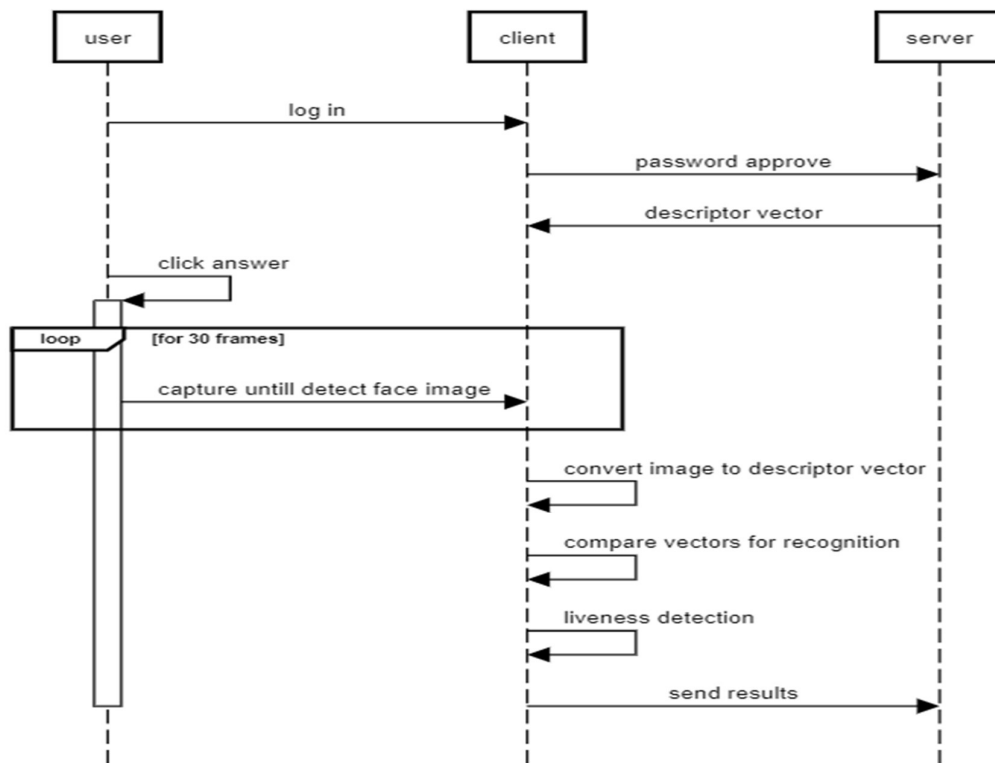
shows the authentication process for one student. These operations are performed for all users at the frontend of each of them.



System architecture

1.3. SEQUENCE DIAGRAM

The following sequence diagram shows the functionality of the system and the interaction of organized objects in time sequence. It describes the sequence of messages between the objects in the scenario and the classes and the items necessary to perform the function of the scenario.



Sequence diagram of the system

1.4. SYSTEM PARTS

In this section, we will describe in details the implementation of the different parts that constitute the system and how they save the data of the students like (id, image, descriptor, etc..) and where the exam and the operation will be done.

1.4.1. SYSTEM DATABASE

In our database we have three tables:

- ❖ Table Students used for student login, that has the following attributes:
 - Sid: Students ID.
 - Name: Students Name.

- Email: Students Email.
- Password: Email's Password.

+ Options

				Sid	name	email	password
<input type="checkbox"/>	Edit	Copy	Delete	4150	mohamd	4150@gmail.com	4150
<input type="checkbox"/>	Edit	Copy	Delete	4151	jorj	4151@gmail.com	4151
<input type="checkbox"/>	Edit	Copy	Delete	4152	nancy	4152@gmail.com	4152
<input type="checkbox"/>	Edit	Copy	Delete	4153	ali	4153@gmail.com	4153

☐ Check all
 With selected:
 Edit
 Copy
 Delete

Table Students

❖ Table Descriptors that has the following attributes:

- ID: Auto Increment Primary Key.
- Sid: Students ID.
- Image: Refers to image name.
- Time: Time when image captured.
- Descriptor: Is a vector of values, which describes the image patch around an interest point. (used for face recognition)

				ID	Sid	image	time	descriptor
<input type="checkbox"/>	Edit	Copy	Delete	1	4153	1.png	Mon May 31 2021 18:22:35 GMT+0300 {"0":-0.160901740193367,"1":0.17409582436084747,"2...	(Eastern Europea
<input type="checkbox"/>	Edit	Copy	Delete	2	4150	2.png	Fri Jun 04 2021 23:15:34 GMT+0300 {"0":-0.13453181087970734,"1":0.1873743087053299,"...	(Eastern Europea
<input type="checkbox"/>	Edit	Copy	Delete	3	4150	3.png	Fri Jun 04 2021 23:16:52 GMT+0300 {"0":-0.12427718937397003,"1":0.195871964097023,"2...	(Eastern Europea










☐ Check all
 With selected:
 Edit
 Copy
 Delete
 Export

Table Descriptors

❖ Table Results for saving the authentication results. It has the following attributes:

- ID: Auto Increment Primary Key.
- Sid: Students ID.
- Recognized:
 - True: Recognized that the captured face image corresponds to the logged in student.
 - (0.xy): Percent of Recognition match in decreasing order.
 - False: Recognized that is other person than logged in.
- Live:
 - Real: Real person
 - Fake: Spoofing detection.
- Time: Time when image captured.

+ Options

				ID	Sid	recognized	live	time
<input type="checkbox"/>	 Edit	 Copy	 Delete	10	4150	true (0.31)	fake	Fri Jun 04 2021 23:36:11 GMT+0300 (Eastern Europea
<input type="checkbox"/>	 Edit	 Copy	 Delete	12	4153	true (0.4)	real	Tue Jun 08 2021 21:05:10 GMT+0300 (Eastern Europea
<input type="checkbox"/>	 Edit	 Copy	 Delete	16	4153	true (0.36)	fake	Tue Jun 08 2021 21:05:43 GMT+0300 (Eastern Europea





☐ Check all With selected:  Edit  Copy  Delete  Export

Table Results

1.4.1.1. ADDING IMAGES AND DESCRIPTORS

University agent may add images and descriptors to database or delete from it, for a student by searching for his ID, to have table as shown below.


ID	Name	Image	Operation
4153	ali		DELETE
NEXT			

Table of student images

The agent searches for a student by his/her ID, where it is assumed the IDs are available in database relation Student.



Capturing image

Here a video start steaming using webcam for capturing image where when button pressed, the web browser uses “face-api” library in JavaScript to detect face from video frames. After detecting a face, image is cropped, resized to 32x32, and used as input to a deep neural networks that transforms it into a descriptor vector which is a real vector of dimension 128.

The vector is transformed into a json array, as in below figure, which submitted to the server and saved in the database using AJAX.

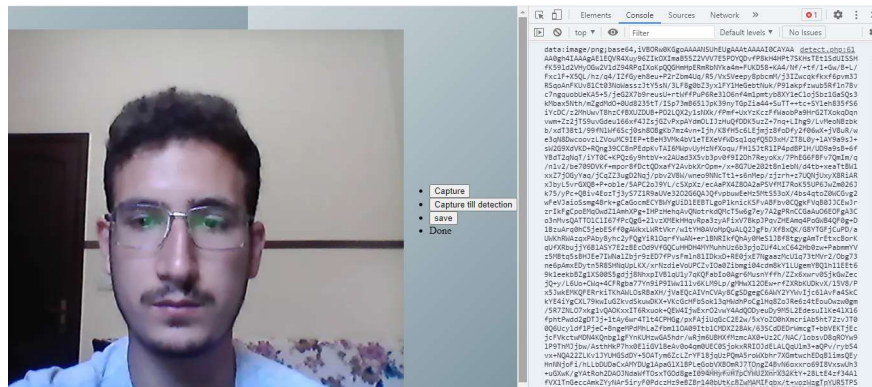


Image and its descriptor

1.4.2. EXAM PART

This part depends on two main functions which will talk about face detection, recognition, and liveness detection.

When a student logs in to take an exam as in below figure, a session is opened, and the descriptor vector of that student is fetched from the database and sent to the client as an element of the web page in hidden input element of a form, to be used then for recognition.

Face detection, recognition, and liveness detection operations take place every time the student answers a question, and the results are submitted to the server.

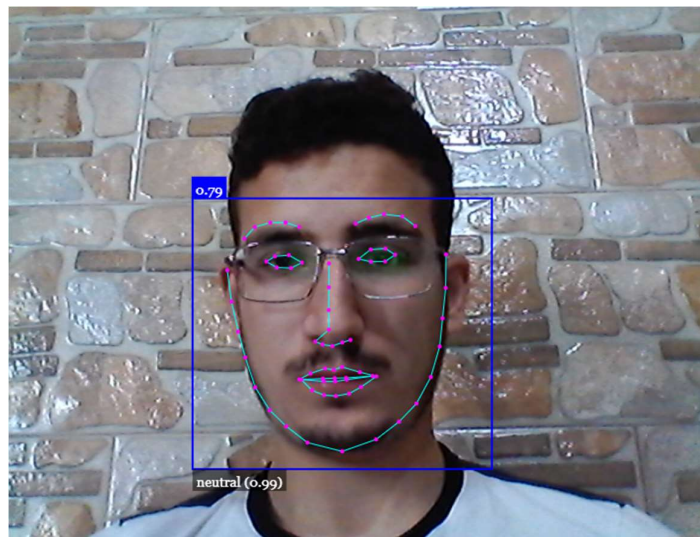
ID:

Password:

Login page

1.4.2.1. FACE RECOGNITION

We use “webcam.js” for opening video and taking a snapshots, and “faceapi.js” for loading trained modules (one model for face detection and one model for face recognition). First we use these modules to detect faces from image and test it as shown below.



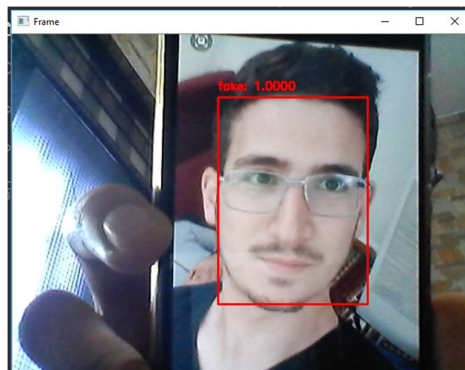
Face detection

Then for face recognition, frames are captured until a face is detected or the number of captured frames reaches 30 frames. Once face is detected, the descriptor vector is computed and compared with descriptor vector of logged in user. We test the code by passing an image, so it successfully worked.

At the end we have a full recognition code where it takes the captured image, detect the face, and then convert it to a descriptor vector. If recognition done it gives the name, else it gives “unknown”.

1.4.2.2. LIVENESS DETECTION

For liveness detection we pass an image of 32x32 size to a trained model to predict if fake or real.



Liveness detection