International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 05 Issue: 09 | Sept - 2021 ISSN: 2582-3930

ONLINE ATTENDANCE USING FACE RECOGNITION

Ritu Gour, Saloni Kothari, Shivangi Dubey, Mrs Prajakta Naregalkar(Assistant Professor)

Department of Electronics, Bharati Vidyapeeth College of Engineering Pune, India

***Abstract*: The advances in technology and some situation and conditions have now enabled the students and professors to take classes from inside their homes. While this is remarkable, it presents an equally challenging problem about the management of the attendance of the students. This paper provides the details about how a mobile phone application can ease this problem by applying a two factor authentication through Bluetooth based IOT device and face detection then verification of the student using a process that will take seconds to complete.**

Keywords: react-native, expo, FaceDetector, Nodejs, MySQL, Microsoft Azure FACE Api, Base64 encoding, raspberry pi.

I. INTRODUCTION

Multipurpose image recognition app is a mobile based app which uses Internet of things by incorporating Bluetooth peripheral using a raspberry pi. This app is designed to provide a minimalistic yet prominent approach for marking and maintaining attendance of the class. This app has also been designed to save time of the class.

II. PROPOSED WORK

It will use internet of things which will include a unique Bluetooth peripheral for each class. This Bluetooth device will be configured to host a server. The Bluetooth USB Id of the Raspberry Pi will be entered in the server. It will be maintained to enable two-way authentication of the students. Students would connect to the Bluetooth device to ensure the first check of the two-way authentication i.e., of the location. The student found to be within that location or area would be allowed to enter into the second phase of authentication. The student would then be permitted to mark the attendance using face recognition in which the face is verified with that stored on the database of Microsoft Azure FACE API. If the face exists then the attendance is marked else the student will be prompted to show face again.

III. METHODS

The image recognition application made by us uses the javascript language based react library's react-native framework, react dom framework and expo framework for the android/ios application.

The teacher's information and detail input panel makes use of the express framework of nodejs to handle the input about the teacher/student details and forward it to the database.

The database is made using SQL with the help of the MySQL workbench tool to create the SQL server.

The in-app facial recognition is performed through the FaceDetector library provided by the expo framework.

***Frontend*:** The frontend of the mobile application was made using Javascript and react-native platforms. The front-end of the Teacher information panel was made using HTML-CSS-JS and then converted to Jade View Engine.

***Backend*:** The backend for the mobile application was handledon the local machine it was used to develop using the default development server provided by the metro bundler of react native development environment. The complete backend and the processes of the teacher input panel were also hosted by the local machine on which they were created (localhost) through nodejs.

First Steps: To create the mobile application we used the commands of the expo-cli in the command window. Then the basic structure of the application was created using pure react-native coding.

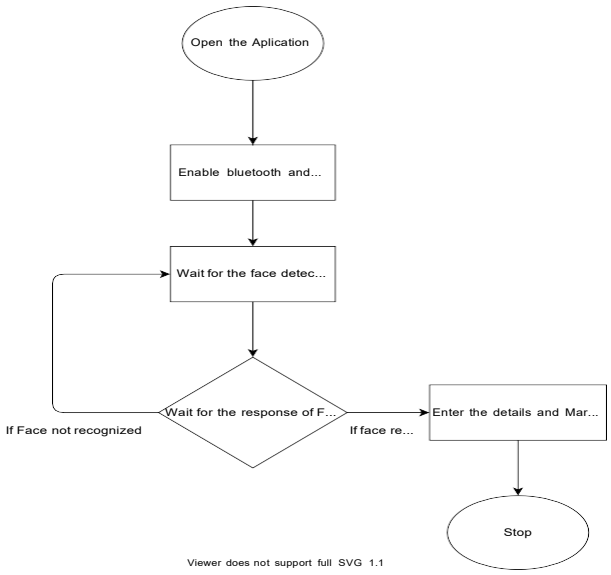
***EXPO CLI*:** It is the main interface for the developer using the expo tools. The list of commands we used:

1. expo start
2. expo install <dependencies>

The react-native code was written using the following libraries of expo:

Permissions, Constants, FaceDetector, axios

***WorkFlow*:**



Like the workflow suggests, the application uses a raspberry pie device as a bluetooth peripheral to verify the location of the student (user). It only proceeds upon verification.

It then uses the camera module to detect the face through the FaceDetector Library internally.

The detected image is then sent to the Microsoft azure servers through the FACE API. The api cannot transport the image as a file thus the image is first cached as a base64 string, then it is decoded using an alternative of the “window.atob()” method to an array buffer.

Then a response from the FACE api is expected. According to the response, it is decided whether to proceed forward (on successful detection of face) or loop back to the previous state (on failure).

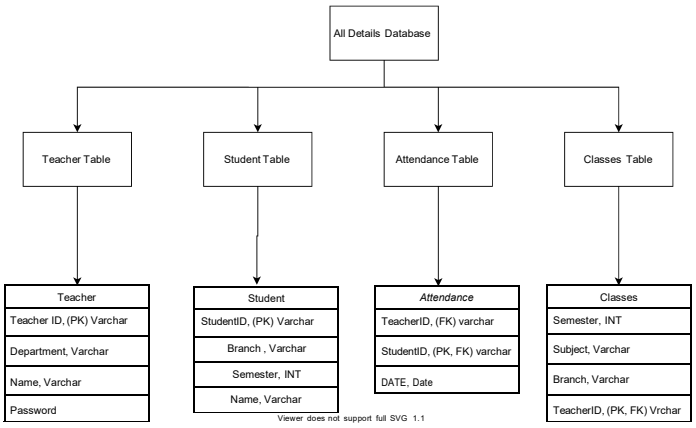
Upon success the student is then asked for the required details of enrolment number and the unique professor identifier so that attendance can be marked.

To handle the marking of attendance and other backend requests like updating teacher information on the SQL database, a Nodejs based server is created using expressjs.

The ports used to run the servers on local machine:

a. 3000: Nodejs Server
b. 19000: Metro bundler development server
c. 3306: MySQL Server

The MySQL database used to store all the information had the following schema:



IV. RESULTS

The face detection response came directly through the FACE api in the following json format:

```
face detect res:  Array [
  Object {
    "faceId": "abba62d2-1c91-4d24-a41b-4ec4e52a405e",
    "faceRectangle": Object {
      "height": 2308,
      "left": 657,
      "top": 2115,
      "width": 1704,
    },
  },
]
```

This is an Array of detected faces in the api.

***faceId*:** It is the unique id given to every face.

***faceRectangle*:** Details about the shape of the face.

The faces with these details are then then checked if they are present in existing database of faces. Following is the data returned if it is present:

```
find similars res:  Array [
  Object {
    "confidence": 0.676065743,
    "persistedFaceId": "882e52d2-1132-427d-bb45-6a9a9e614a4e",
  },
]
```

Confidence: This property tells the confidence in the detected face. In above pic, the confidence is 67.6%.

***Persisted FaceId*:** the face id of the face in database with whichit was actually matched.

Out of 13 trials, I got results with:

Confidence below 40% : in 3 trials.

Confidence between 40-80%: in 5 trials.

Confidence above 80%: in 2 trials.

No.	Success/Failure	Confidence	Time (s)	Reason
1	Success	62%	13	-
2	Success	43.5%	12	-
3	Success	73.43%	14	-
4	Failure	0	23	Not Register
5	Success	54.2%	18	-
6	Success	14.34%	11	-
7	Success	82.93%	14	-
8	Success	64.97%	14	-
9	Success	19%	16	-
10	Failure	0	33	Low Light
11	Success	19.33%	21	-
12	Failure	0	31	Low Light
13	Success	91.011%	13	-

Number of successes: 10 / 13, 76.9%

Number of failures: 3 / 13, 23.1%

Average time for the process to complete (for success): 14.6 s.

The most probable cause of failure: Picture taken in low light.

V. DISCUSSION

This app has been designed to save time of the class while marking the attendance. Usually we see that in an hour class 10-15 minutes are usually gone in taking attendance and counting the absentees. This app thus provides a simpler and time efficient way of doing the same. Also, it prevents the chances of proxy marking by providing two way authentication which incorporates location and face recognition. It serves being the Multipurpose by providing teachers the use of a database where they could easily access the day to day records of the students and the students can also keep track of their attendance.

VI. CONCLUSION

This app thus provides an eminent way of marking and maintaining attendance of a class. This method not only saves paper but also saves time of the class by making the best use of the technology. This app provides students with the data of their attendance so that they can keep track of it. The Teachers also get more time to teach and can keep record of student's day to day attendance.

VII. ACKNOWLEDGEMENT

Preparing any project requires views of many people but we suspect that technical projects are more different than others. A great many people are involved in various stages of the project preparation who shared their knowledge and experience.

Before we get into thick of things, we would like to add a few heartfelt words for the people who are part of this project in numerous ways people who gave un-ending support right from the stage idea was conceived.

It is our proud privilege to express a deep sense of gratitude and regard to our Project Guide Prof. Amit kumar Manjhawar (Asst. Professor, Dept. Of Computer Science & Engineering and IT) for the opportunity, he has given us for carrying out the project. His initiative, keen interest, expert guidance at every step provided a constant source of inspiration and encouragement to us for intense studies in the subject. We are deeply indebted to him.

Our sincere thanks to all those who directly or indirectly helped us in the development and evolution of thoughts.

VIII. REFERENCES

[1] https://docs.expo.io/
[2] https://reactnative.dev/docs/getting-started
[3] https://nodejs.org/en/docs/
[4] https://github.com/noble/bleno
[5] https://github.com/mathiasbynens/base64
[6] https://www.npmjs.com/package/axios
[7] https://github.com/axios/axios
[8] YangQun Li, An Integrated Platform for the Internet of Things Based on an Open Source Ecosystem, IEEE Journal for future Internet.
[9] Gopinath Shanmuga Sundaram, Bhanuprasad Patibhandala, Bluetooth communication using touchscreen interface with the Raspberry pi, 2013 Proceedings of IEEE Southeastcon
[10] Getting Started with the Internet of Things: Connecting Sensors and Microcontrollers to the Cloud, Cuno Pfister, 2011
[11] https://docs.expo.io/versions/latest/sdk/facedetector/
[12] https://developers.google.com/vision/face-detection-concepts
[13] Digital image processing, 2/E, Richard Gonzales Richard Woods