AUTOMATED ATTENDANCE MANAGEMENT SYSTEM

Submitted in partial fulfillment for the award of the degree of

Bachelors of Technology in Computer Science and Engineering

By

Sujay (185001174)

ABSTRACT

Legacy attendance systems/practices involve the active participation of both the participant and the teacher/presenter taking a roll call of the participants. This consumes a large proportion of the time meant for the lecture/presentation. An automated attendance system eliminates the need for the participation of the lecturer/presenter in the tedious attendance taking process. The system aims to digitize and modernize the archaic process of taking attendance by hand.

This system once implemented can be used to automate the entire attendance process by just taking an image or uploading an image which is already captured and then award the attendance to everyone present in the image.

CONTENTS

INDEX

LIST OF FIGURES

LIST OF ACRONYMS

API - Application Programming interface

CI/CD - Continuous Integration and Continuous Development

CNN - Convolutional Neural Network

CSS - Cascaded Style Sheets

DB - Database

HTML - Hyper Text Markup Language

JS - JavaScript

UI - User Interface

UX - User Experience

YOLO - You Look Only Once

NON-PAT INTERNSHIP

Goodera is a state of the art technology platform that helps companies measure and communicate impact and ROI of CSR, Sustainability and volunteering initiatives. Currently, 200+ corporates (24 in Fortune 500) including P&G, Target, Dell, GAP, Abbott, Amazon are using the Goodera platform across 90+ countries to track, monitor and measure 250mn+ development capital annually to 6mn+ beneficiaries. Goodera enables companies to collect data from multiple sources including web, mobile, voice, API integrations with internal tools and have real-time dashboards to communicate CSR & Sustainability performance to stakeholders like CEO, Board, Employees, Investors etc. and report as per external global standards- GRI, SDGs, DJSI, CDP, IR amongst others. Goodera is amongst the first VC backed companies in this space with Nexus Venture Partners and Omidyar Network as key investors. Goodera has a 120+ team with strong domain and technology expertise.

- Goodera Enterprise: One stop platform (for Corporations, Foundations, Governments) to manage their grants, measure the impact they create by providing data-driven comprehensive insights along with stories of change in a real time basis.
- Goodera Volunteer: One stop platform for employees of the corporates to discover volunteering events across several social cause areas across multiple geographies at scale.
- Goodera Partner: Goodera partner is a unique platform for non-profits where they are able to showcase their programs or volunteering opportunities for corporate employee volunteers and increase their organisation’s visibility among corporates.

MY ROLE AND RESPONSIBILITIES:

- Interned in the Product team and worked with developers, designers, and product managers.
- Created dashboards with various data such as NPS of the entire enterprise product, various platform analytics.
- Setup various analytical flows in the platform to gather user feedback data
- Implement various fix to reduce pain points by the clients and increase the NPS score

Chapter 1

Introduction

In a classroom where attendance is taken every hour, it consumes a minimum of 10% of the total class time which is 5 minutes on an average of 50 min for the classes. While taking attendance is crucial for every class, the time taken can however be reduced drastically with the help of technology.

With the advancement in technology being applied to all the fields,
Our quality of life has improved making an entire world of internet available in our palms, despite these advancements few things have remained in the same old fashioned way and taking attendance, thus there is a pressing need make the whole task of taking attendance more efficient and easier.

The main aim of this project is to upgrade the existing attendance and utilize the technology fasten up the entire process to ensure class time is not wasted on thins which is as basic as an attendance.

1.1 OBJECTIVES

The main aim of this project is to upgrade the existing attendance infrastructure with existing technologies to improve the time efficiency and reduce the possibilities of proxy and errors.

The objectives are

- Process the idea of the newly proposed system.

- Construct flows for teachers, students, and an admin

- To optimize the product and make it as robust as possible

- Analyze the products flaws, disadvantages, and weaknesses.

1.2 BACKGROUND

The current form of attendance taking is a time-consuming, slow and a medieval process. It set out to begin with a record-keeping task which evolved to a rather cumbersome task due to the amount of time consumed and the amount of concentration required by both the parties to make it a smooth process.

Even with advancements in technology and ease of access to the advances in tech, the method of taking attendance has remained the same and has seen no advancements or improvements. Thus the there is a pressing need to upgrade the entire way attendance is taken which can help in making it more efficient and easy. This has to be achieved by implementing various technologies that has become standard part of everyone’s day-to-day life such as mobile phones, cameras and the internet.

Chapter 2

Proposed System Architecture

2.1 PROPOSED DESIGN

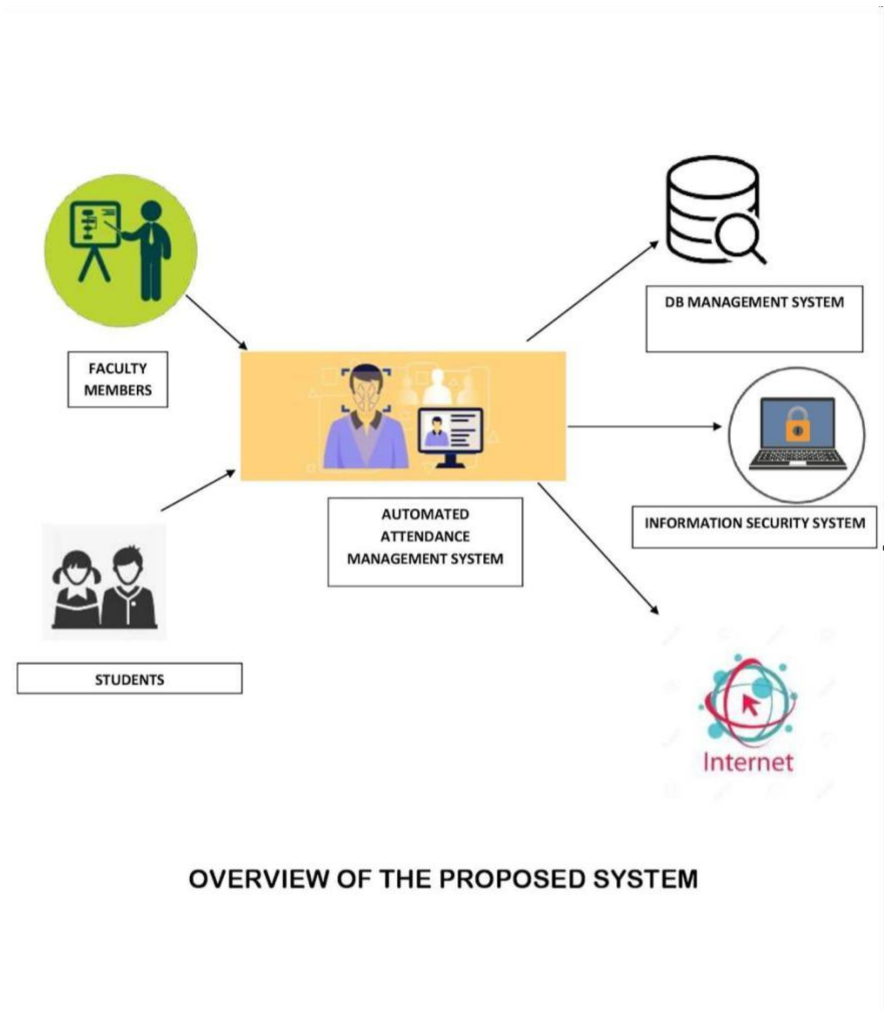

FACULTY MEMBERS
STUDENTS
AUTOMATED ATTENDANCE MANAGEMENT SYSTEM
DB MANAGEMENT SYSTEM
INFORMATION SECURITY SYSTEM
Internet
OVERVIEW OF THE PROPOSED SYSTEM

Fig 2.1 : Proposed System overview

2.2 PROCEDURE

The system design in figure 2.1 showcases the various parts of the entire system which coupled together makes the automated attendance management system.

The students will need to sign up for the platform when using it for the first time by providing various details like registration number, name, number of subjects and also they will need to click a picture of themselves which will be later used to match when teacher or the admin clicks uploads the picture into the DB. All of this data collection process is done with the help of a simple self-sign up flow on the UI using which the student can navigate easily. Once the initial sign up process is done, whenever the student logins into the platform he will be able to access a dashboard which shows all the classes he has and the attendance percentage along with mean and standard deviation of the entire class which gives the student an easy visualisation of his attendance and comparison with his peers.

The Faculty when logged into the portal will be able to view the classes they are taking and will be able to see the attendance of all the roll numbers present inside her classroom.

The admin when logged into the portal will be presented three options, he has an update timetable option in case the time table of one particular day is changed, and also manually check and confirm the time table of that particular day. Once he clicks on the update timetable button the classes for that particular day are updated for all the students present in the classes. The second option is to capture the attendance by uploading an image or capturing an image using the inbuilt camera of the device, this option will use the image as input and then use various models and algorithms explained in the upcoming section of this thesis to recognise the faces and award attendance to everyone present in the images.

Chapter 3

LITERATURE SURVEY

3.1 LEARNING FEED-FORWARD ONE- SHOT LEARNERS

Usually, generative models or discriminative embeddings are used to deal with one-shot learning. Deep learning-based discriminative algorithms, which are very effective in other learning contexts, are not well suited for one-shot learning since they require a huge amount of training data. We offer a method for learning the parameters of a deep model in a single shot in this paper. The learner is built as a learnet, which is a second deep network that predicts the parameters of a student network from a single instance. By minimising a one-shot classification objective in a learning to learn formulation, we can construct an efficient feed-forward one-shot learner that can be trained end-to-end. We offer a number of factorizations of the pupil network's parameters in order to make the construction practical. We show promising results in the Omniglot benchmark by learning letters from single exemplars and in the Visual Object Tracking benchmark by tracking visual objects from a single initial exemplar.

3.2 FACE RECOGNITION - A ONE-SHOT LEARNING PERSPECTIVE

The ability to learn from a single instance is unique to humans, and one-shot learning algorithms attempt to replicate this skill. On the other hand, despite the excellent performance of Deep Learning-based approaches on a variety of picture classification problems, performance is frequently dependent on the availability of a large number of annotated training samples per class. This is unquestionably a barrier to deploying deep neural network-based systems in many real-world applications, such as facial recognition. Furthermore, adding a new class to the system will necessitate retraining the entire system from the ground up. However, the power of deep learned features must not be overlooked. The goal of this study is to combine the advantages of deep learnt features with a classic One-Shot learning architecture. The results from two publicly available datasets are quite promising, with over 90% accuracy on 5-way One-Shot jobs and 84 percent accuracy on 50-way One-Shot challenges.

3.3 OVERVIEW OF THE FACE RECOGNITION GRAND CHALLENGE

Face recognition researchers have been working on new algorithms over the past few years. Advances in computer vision techniques, computer design, sensor design, and interest in fielding face recognition systems are all fueling these breakthroughs. Such advancements have the potential to reduce the mistake rate in facial recognition systems by an order of magnitude when compared to the results of the Face Recognition Vendor Test (FRVT) 2002. The Face Recognition Grand Challenge (FRGC) is a six-experiment challenge problem with a data corpus of 50,000 photos that is aimed to attain this performance target. 3D scans and high-resolution still pictures obtained under controlled and uncontrolled situations make up the data. The challenge problem, data corpus, baseline performance, and preliminary results on natural statistics of facial imaging are all described in this work.

3.4 FACENET: A UNIFIED EMBEDDING FOR FACE RECOGNITION AND CLUSTERING

Despite recent major breakthroughs in the field of face recognition, deploying face verification and recognition at scale poses substantial challenges to current methodologies. FaceNet is a system that learns a mapping from face images to a compact Euclidean space in which distances directly correspond to a measure of face similarity. Face recognition, verification, and clustering may all be done using traditional approaches with FaceNet embeddings as feature vectors once this space has been created. Rather than an intermediate bottleneck layer, as in prior deep learning systems, the method use a deep convolutional network trained to directly optimise the embedding. We employ triplets of roughly aligned matching / non-matching face patches generated by a unique online triplet mining approach to train. Our method has a significant advantage in terms of representational efficiency: we can obtain state-of-the-art face recognition performance with only 128 bytes per face. Labelled Faces in the Wild (LFW) dataset, which is widely utilised The system sets a new high of 99.63 percent accuracy. It scores 95.12 percent on YouTube Faces DB. On both datasets, the technique reduces the error rate by 30 percent when compared to the best published result.

3.5 BLUETOOTH BASED ATTENDANCE MANAGEMENT SYSTEM

This paper focuses on taking attendance more efficiently. Basically, in this paper an application software is built wherein it is installed in the instructor’s mobile phone where it enables the teacher to monitor/query students device via Bluetooth and through the transfer of the MAC address of the student's device, it confirms the presence of the student where a detailed record can be generated later on if required.

Based on the Security requirements we can add other biometric devices like a fingerprint reader or iris scanner.

3.6 DEVELOPMENT OF ATTENDANCE MANAGEMENT SYSTEM USING BIOMETRICS:

This paper focuses on improving the time taken for manually taking attendance by introducing a system that takes attendance electronically with the help of a fingerprint scanner and the records of the attendance are stored in the database. Each and every time the attendance is marked only after student identification. In student identification, the first time the users are said to register their fingerprints to the admin where he collects the information and stores the information in a database. The next time when attendance is taken when the fingerprint is scanned it checks with the database for a match and then fetches those particular details so that attendance can be marked for the student. According to the paper, there was a significant improvement compared to the manual attendance by around 80%.

3.7 STUDENT ATTENDANCE MONITORING AT THE UNIVERSITY USING NFC:

Students nowadays don’t attend classes regularly as each and every content taught to them is available on the Internet there is a decreasing rate of successful exams. Then they decided to monitor the student’s attendance at the lectures with the help of NFC. This system uses NFC as its authentication system to get the data from the card or the mobile device where the information is preloaded and when either of the two is tapped the attendance is posted to the database.

3.8 IOT BASED ATTENDANCE SYSTEM:

Fingerprints are unique and can be used to identify a person uniquely. Attendance is generally taken verbally and a person needs to verify whether the said attendance is correct manually, so instead of the existing system, they are using these fingerprint scanners to identify a person and send when the person scanned his fingerprint to the webserver Implemented We deploy some Fingerprint sensors, these fingerprint sensors are used identify a person. When a person scans his fingerprint on this sensor, the data gets sent to the webserver using Arduino and the web server can generate a CSV along with when the attendance was punched. Main take-away This paper brings the concept of attendance to the internet with the help of IoT, hence making the process not bound to the legacy way of roll-calls and eliminating time wasted.

The concepts discussed above provide an overall outlook of the current state of the art models for an attendance system that has various approaches towards addressing issues such as speed, efficiency, accuracy, and coordination. Given today’s technological scenario, biometrics coupled with IoT has time and again proven to be the better alternative to traditional systems. Instead of the biometrics discussed in most of the above papers, face detection has proven to be way more secure and can be integrated into the latest mobile phones, which is commonplace among students of our generation. The proposed system aims at not only tackling those issues but optimizing them to the maximum extent possible with the help of IoT principles and tools. In this manner, we implement a system that's fool proof, highly secure and extremely fast in its execution of implementing an automated attendance system, that can give any state of the art concept or implementation a run for its money.

Chapter 4

EXPERIMENTATION

I tried to implement the entire project in various methods using an embedded system which can do the entire process from capturing an image to awarding the attendance on board. These failed models led to the final model which will be presented in detail in the PROPOSED section.

4.1 Attendance System V1

Attendance system V1 was built using with plans to use a raspberry pi as an embedded system to handle all the tasks from capturing the image to DB operations to facial recognition.

The faculty, students and the admin would have the option to connect to the raspberry pi over the local network and perform the operations removing the need to have dedicated servers or cameras to complete the attendance process. This lead to a more independent system which could handle the entire process without depending on any external factors such as a need for imaging or computing equipment.

The system was simple to use, however it had many disadvantages. It was not economical to setup and scaling the system at a later stage would have been impossible because of the choice of architecture used in using the database.

Advantages:

- The students and faculty can access the module from any platform which has a browser and is connected to the local network.

- Can be opened to external networks too, wallowing access to the DB over the internet.

- The system would have been easily scalable as multiple such systems can be integrated together into the same database and inference can be drawn from the same.

Disadvantages:

- The initial version was made using SQL lite, this meant it could handle only a limited number of connections and scaling would have been impossible.

- In this setup, the teacher would have needed to carry raspberry pi along with them to the classrooms in order to take attendance which makes it less intuitive.

- The cost of implementing this would have been higher because each system would needs its own raspberry pi and camera modules to function.

- The image quality of the image captured using the raspberry pi camera module was not up to the mark and the final output was not having high accuracy.

Chapter 5

USE CASE DIAGRAMS

5.1 Main use case diagram

The system has 3 user classes who have access in the system, The following diagram shows how each user can interact with the various actions in the platform.

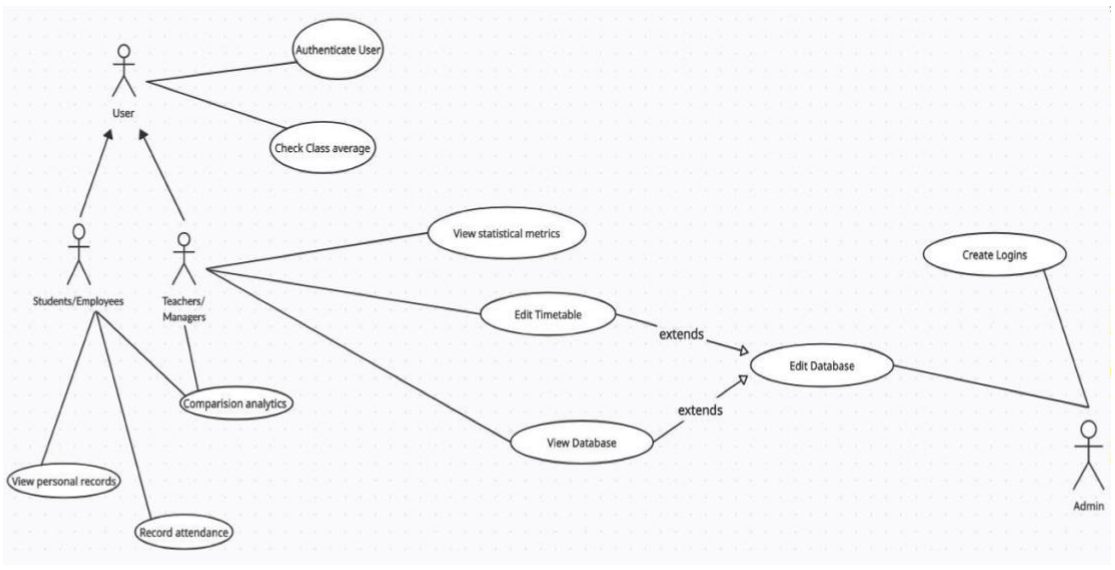


Fig 5.1 Main Use Case Diagram

5.2 Login-State Diagram

This following diagram gives an over-view of the login module and how it is going to function.

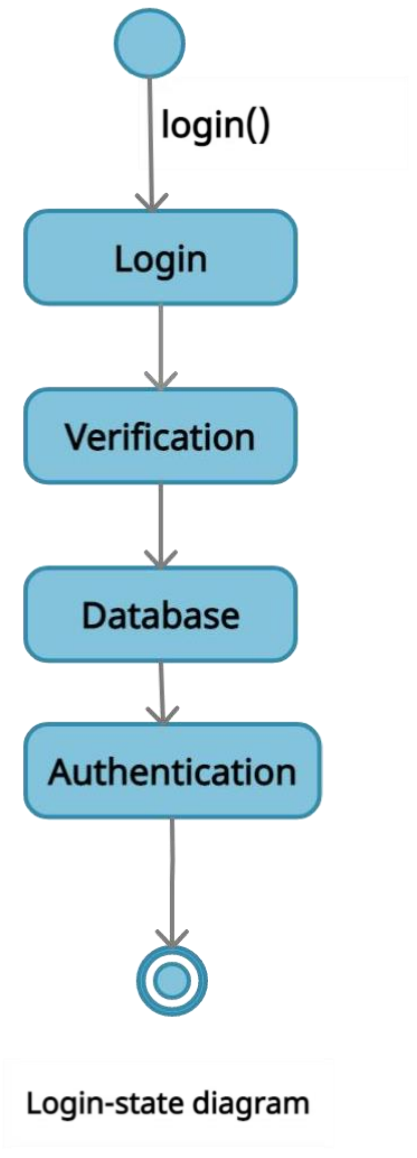


Fig 5.2 Login State Diagram

5.3 Student / employee access - state diagram

The following diagram shows the basic overview of how the modules work in case a student account accesses the platform.

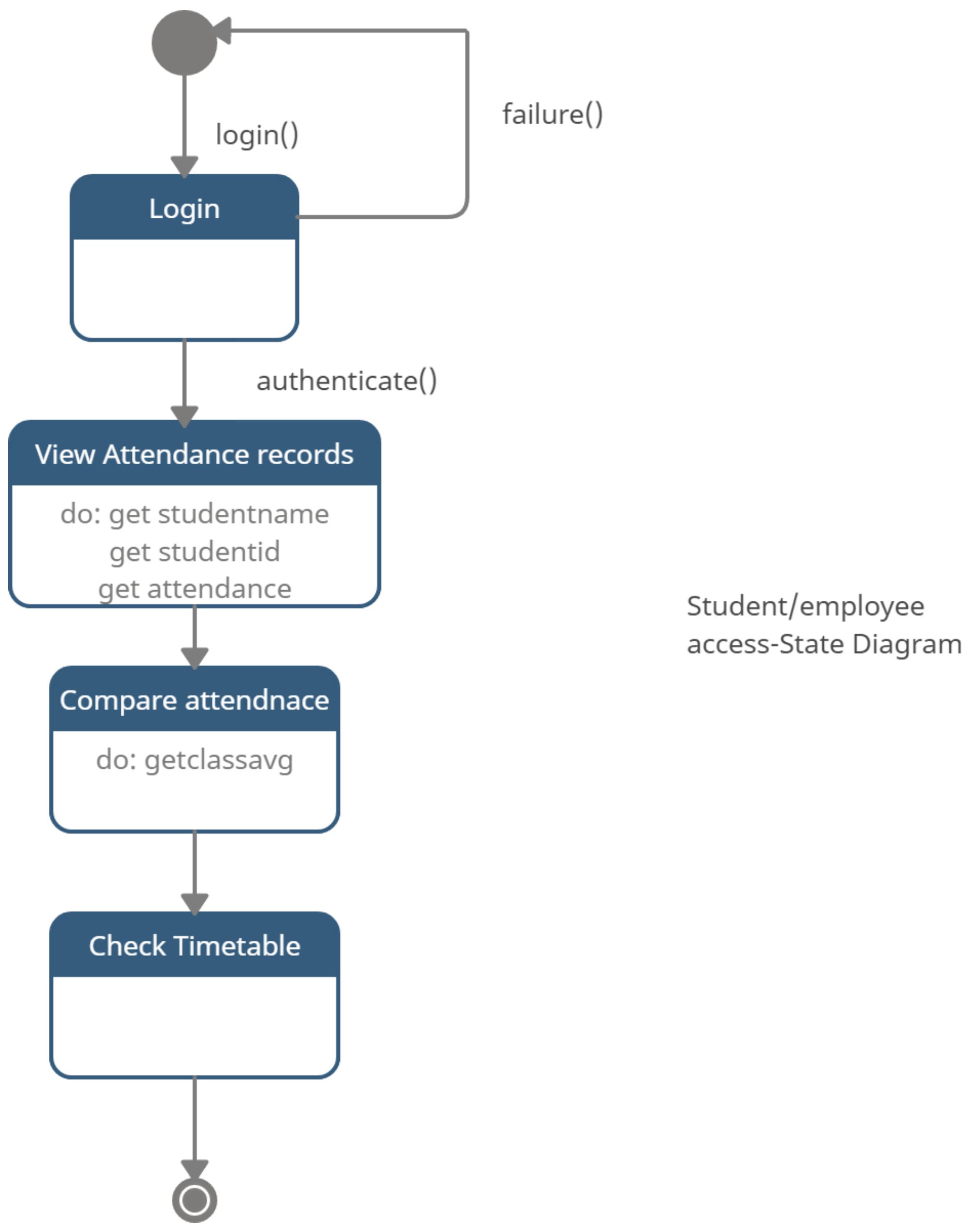


Fig 5.3 Student access state diagram

5.4 Teacher / Manager portal access state diagram

The following diagram shows the basic overview of how the modules work in case a teacher account accesses the platform.

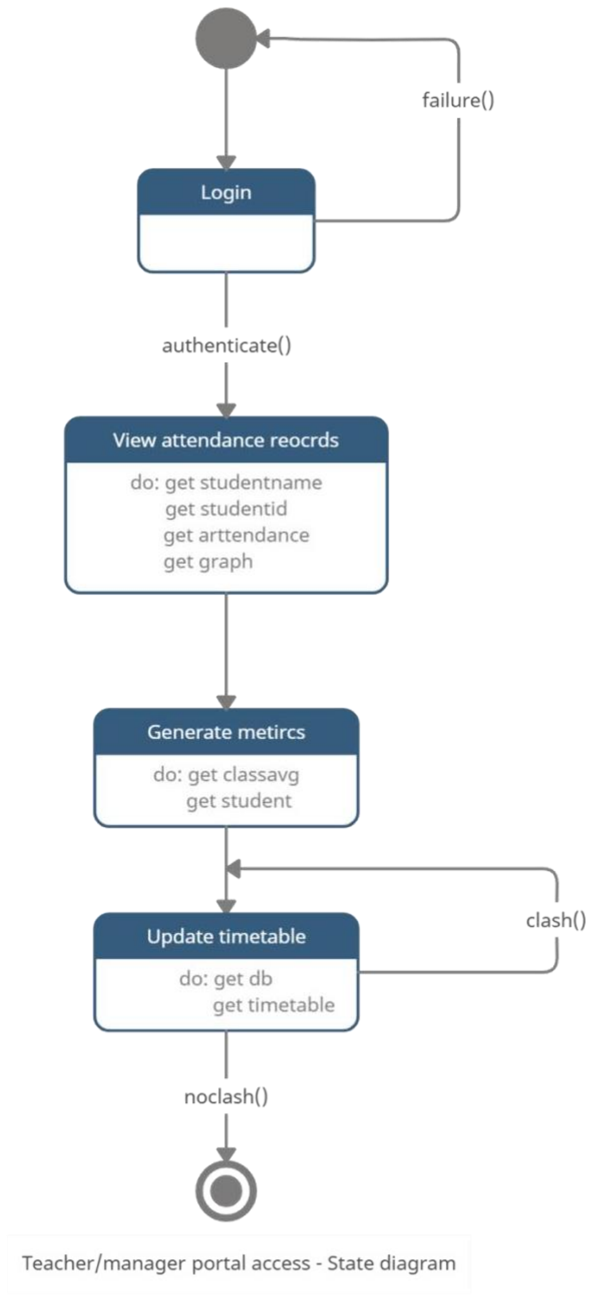


Fig 5.4 Teacher access state diagram

5.5 State transition diagram

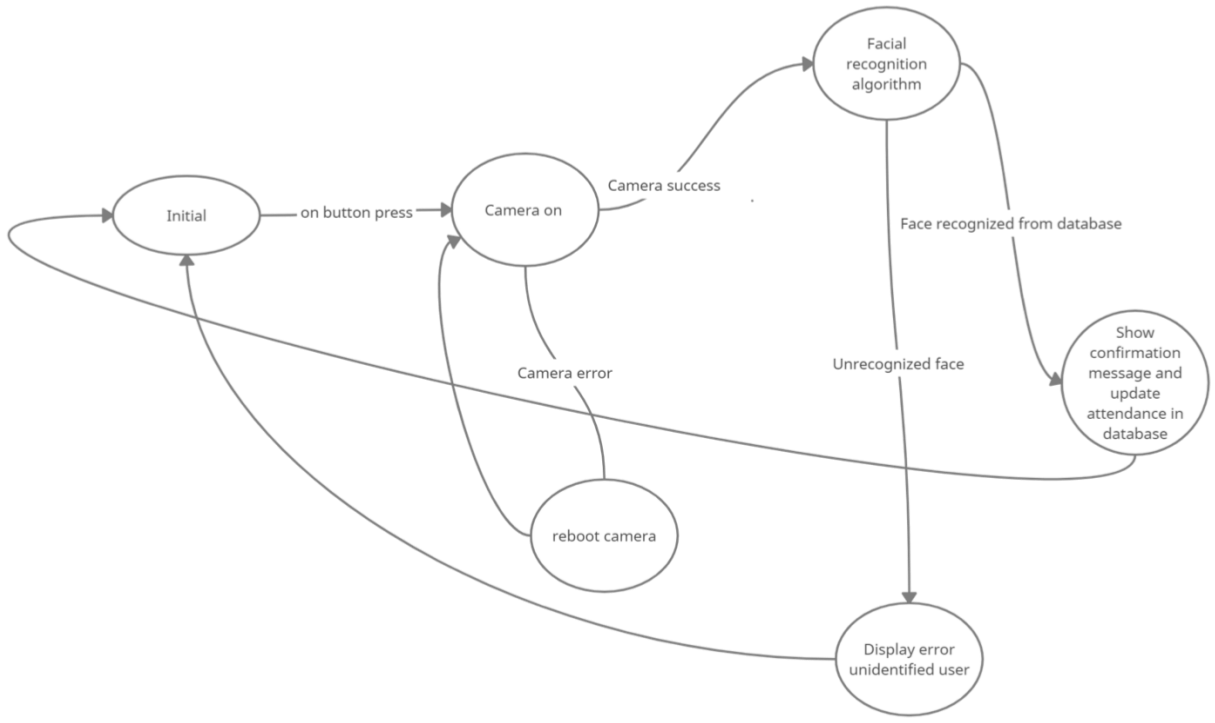

Facial recognition algorithm
Initial
on button press
Camera on
Camera success
Face recognized from database
Show confirmation message and update attendance in database
Camera error
Unrecognized face
reboot camera
Display error unidentified user

Fig 4.5 state transition diagram

Chapter 6

TECH STACK

6.1 Frontend

The Frontend stack predominantly involves technologies like React, HTML and CSS

6.2 Backend

The backend is built entirely using node JS.

6.3Database

MySQL database is used to store student details such as register number name, class id etc in string format. The image of the student is stored in a JPG format. The password is encrypted using MD5 hashing and stored in the database for security.

6.4 Face Detection and Recognition

Using pre trained models such as SSD Mobilenet V1, Tiny Face Detector , Face Recognition model and 68 Point Face landmark detection model which will be explained in detail along with its implementation in the later models.

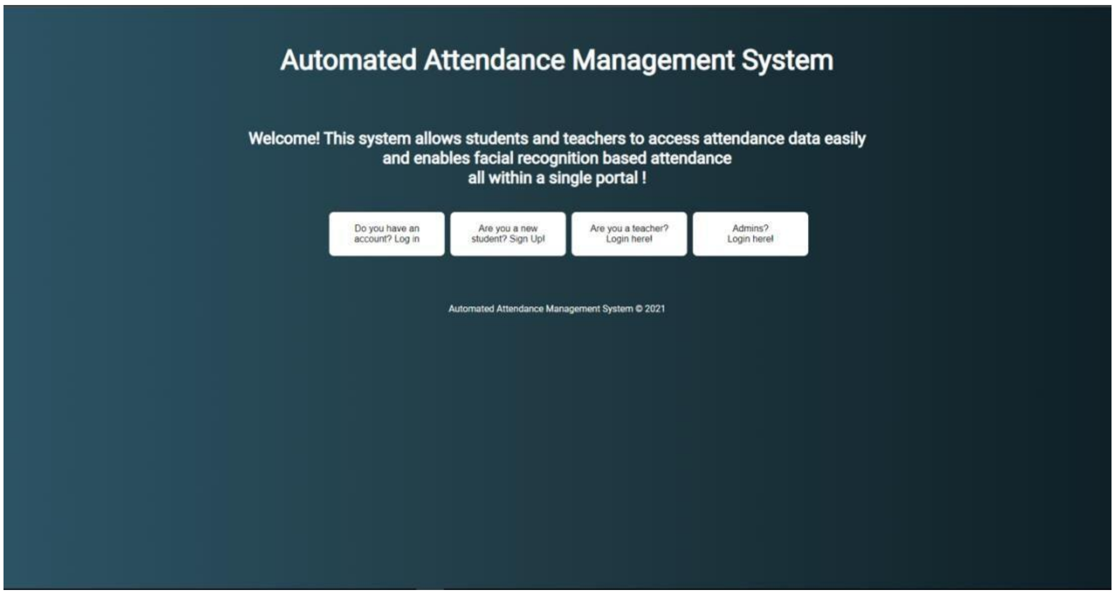
Chapter 7

PROPOSED APPROACH

This final version comprises of various UI/UX changes along with implementation of an complete automated onboarding process for students.

7.1 LANDING PAGE

All the users will be sent to this landing page on clicking the link one the entire webpage is hosted. They are given various options for the various class of users, Admin and Teachers have the option to login while students can create an account for themselves on the platform and sign in using the accounts already present.



Automated Attendance Management System

Welcome! This system allows students and teachers to access attendance data easily and enables facial recognition based attendance all within a single portal !

Do you have an account? Log in
Are you a new student? Sign Up!
Are you a teacher? Login here!
Admins? Login here!

Automated Attendance Management System © 2021

Fig 7.1 : Landing page

7.2 STUDENT APPLICATION

The student application consists of 3 major parts, the sign-up, login and the dashboard pages which are the three major use cases for the student accounts.

7.2.1 SIGN UP FLOW

The user when logging into the platform for the first time will need to complete a self-signup process where in the first screen, they will need to provide with the details such as name, registration number, number of classes and set a password for his account.

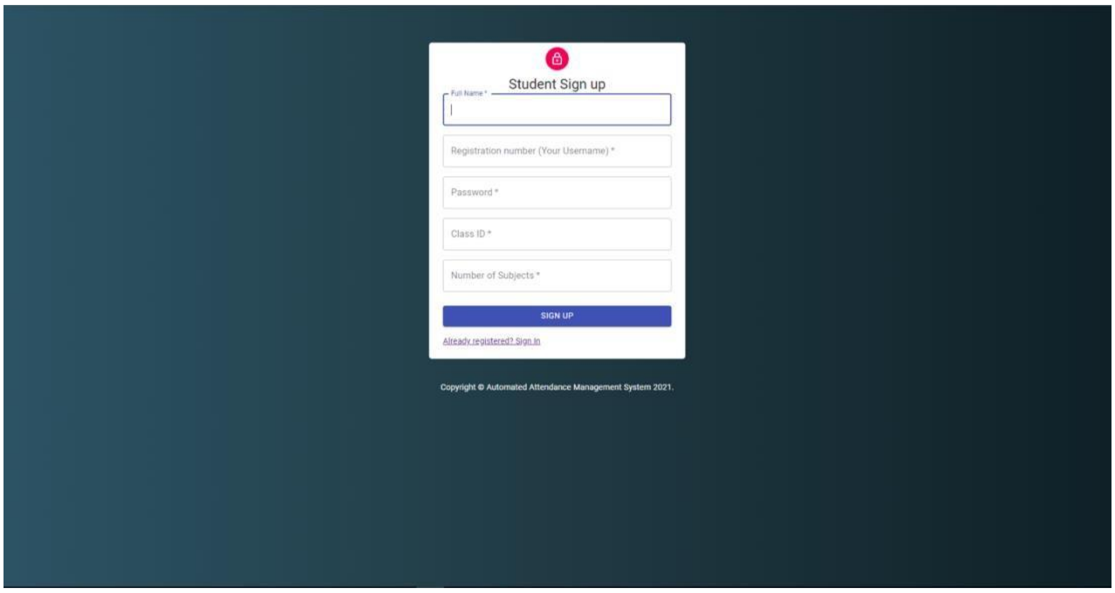
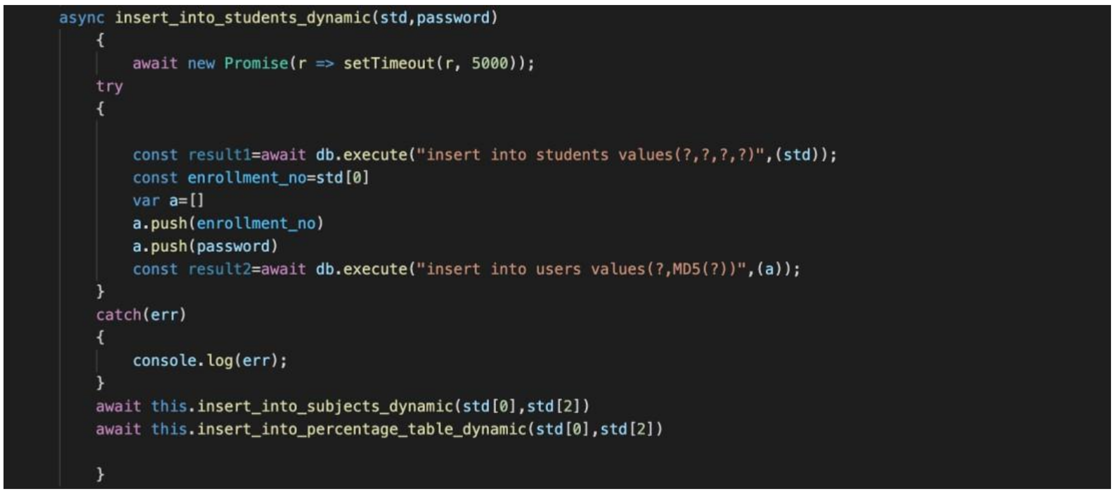

Student Sign up
Full Name *
Registration number (Your Username) *
Password *
Class ID *
Number of Subjects *
SIGN UP
Already registered? Sign In
Copyright © Automated Attendance Management System 2021.

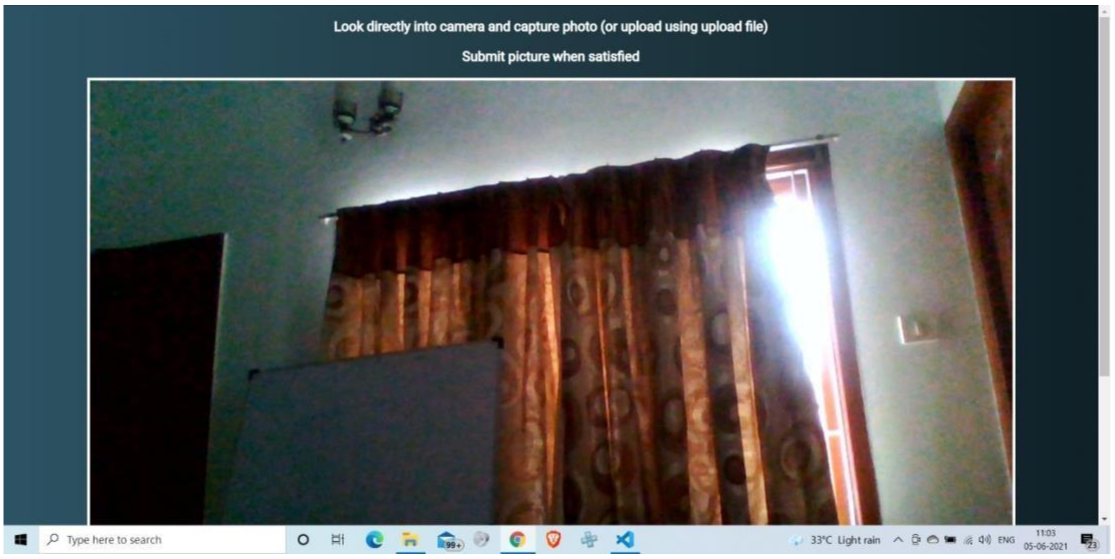
Fig 7.2 : Student Sign up page

The name, registration number and class id is stored in an string format. The image is stored in JPEG format and the password is encrypted using a MD5 hashing and then stored in the MySQL database.



```
async insert_into_students_dynamic(std,password)
    {
        await new Promise(r => setTimeout(r, 5000));
    try
    {

        const result1=await db.execute("insert into students values(?,?,?,?)",(std));
        const enrollment_no=std[0]
        var a=[]
        a.push(enrollment_no)
        a.push(password)
        const result2=await db.execute("insert into users values(?,MD5(?))",(a));
    }
    catch(err)
    {
        console.log(err);
    }
    await this.insert_into_subjects_dynamic(std[0],std[2])
    await this.insert_into_percentage_table_dynamic(std[0],std[2])

    }
```

Once the user fills all the data, they are requested to scan an image using the camera on their device which can later be user to recognize from the image.



Look directly into camera and capture photo (or upload using upload file)

Submit picture when satisfied

Fig 7.3 : Student Sign up page - 2

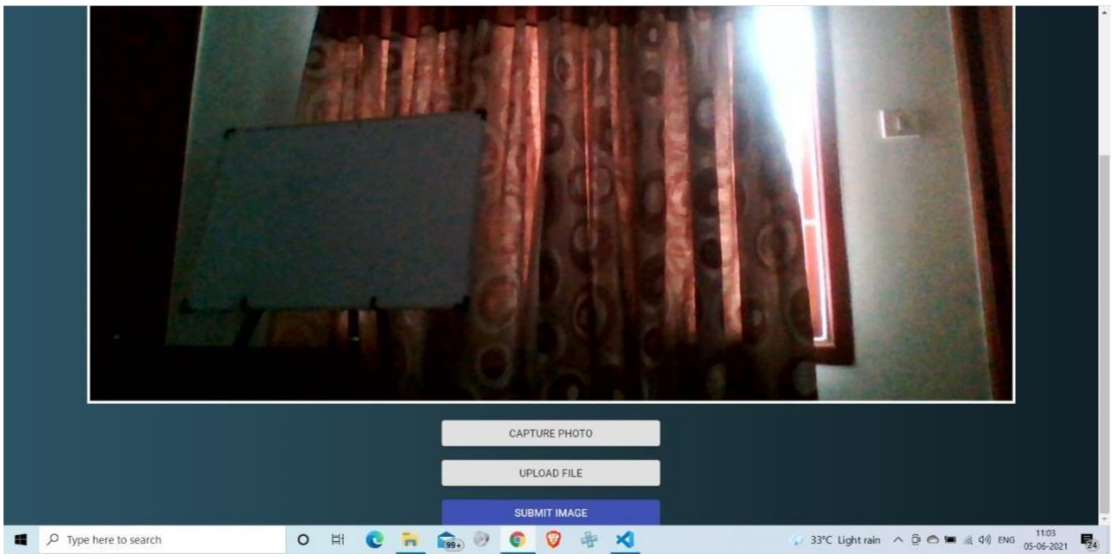


Fig 7.4 : Student Sign up page - 3

This completes the flow from the student. the signup details given by the student is stored in a MySQL database .

7.2.2 Student Login Page

The student once completing the creating of account using the self-signup process can use the signup in the landing page.

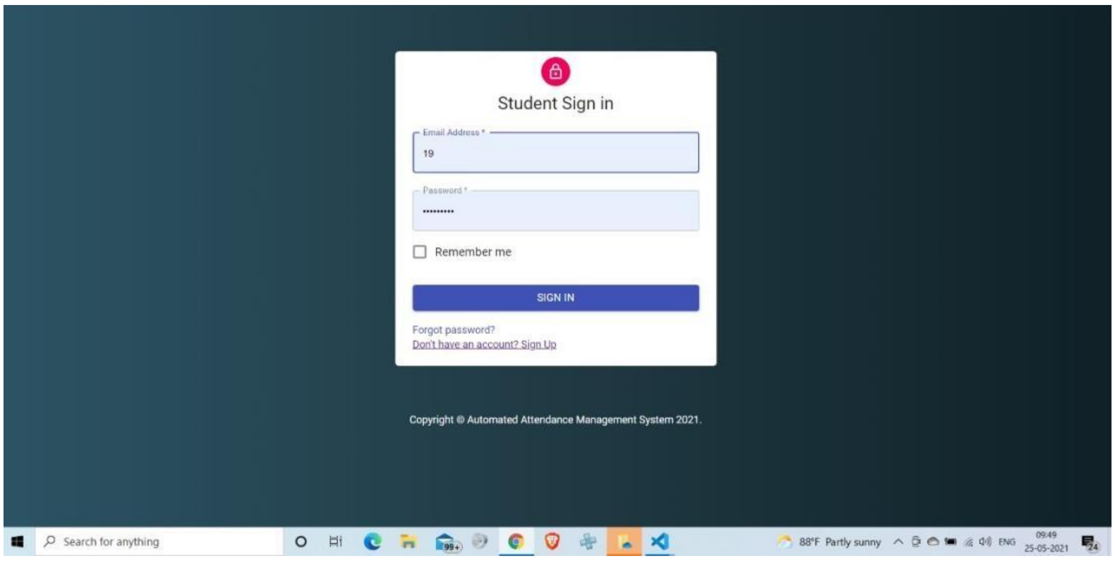
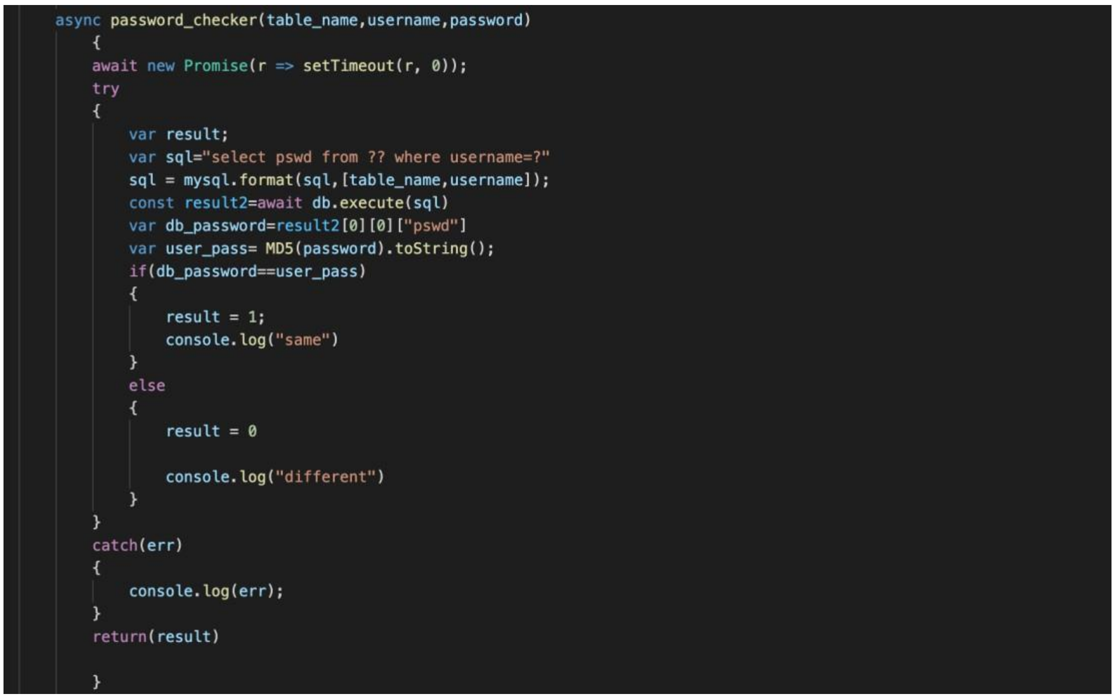


Fig 7.5 : Student Sign in page

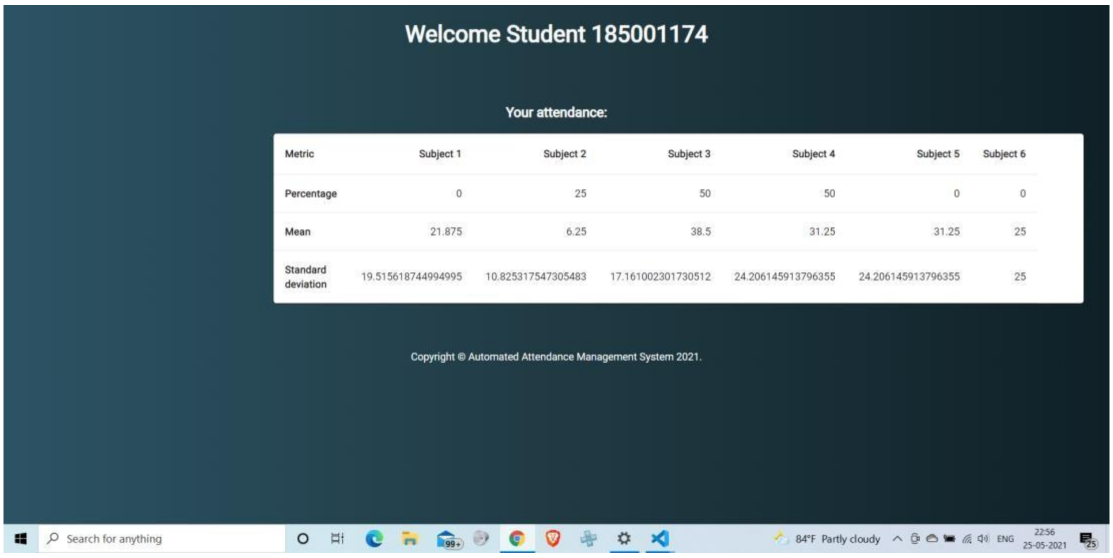
The password user enters is converted using an MD5 hash function and checked with the database.



```
async password_checker(table_name,username,password)
    {
    await new Promise(r => setTimeout(r, 0));
    try
    {
        var result;
        var sql="select pswd from ?? where username=?"
        sql = mysql.format(sql,[table_name,username]);
        const result2=await db.execute(sql)
        var db_password=result2[0][0]["pswd"]
        var user_pass= MD5(password).toString();
        if(db_password==user_pass)
        {
            result = 1;
            console.log("same")
        }
        else
        {
            result = 0

            console.log("different")
        }
    }
    catch(err)
    {
        console.log(err);
    }
    return(result)

    }
```

7.2.3 Student Dashboard

Once the student signs up for the platform, he can login into his dashboard and view his attendance along with the mean and standard deviation for the class he is attending.



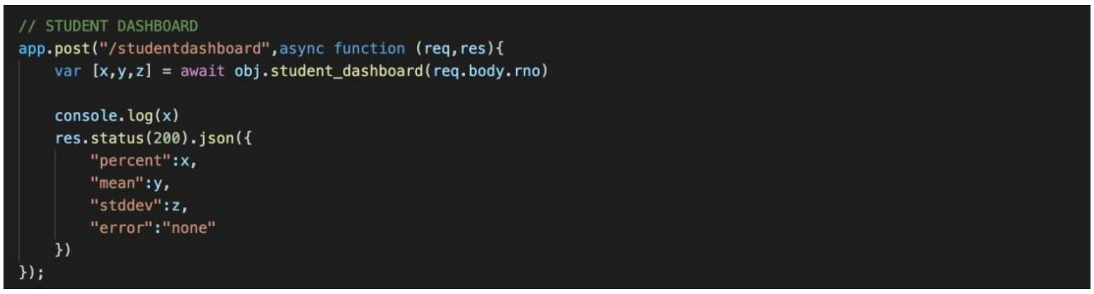
Welcome Student 185001174

Your attendance:

Metric	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5	Subject 6
Percentage	0	25	50	50	0	0
Mean	21.875	6.25	38.5	31.25	31.25	25
Standard deviation	19.515618744994995	10.825317547305483	17.161002301730512	24.206145913796355	24.206145913796355	25

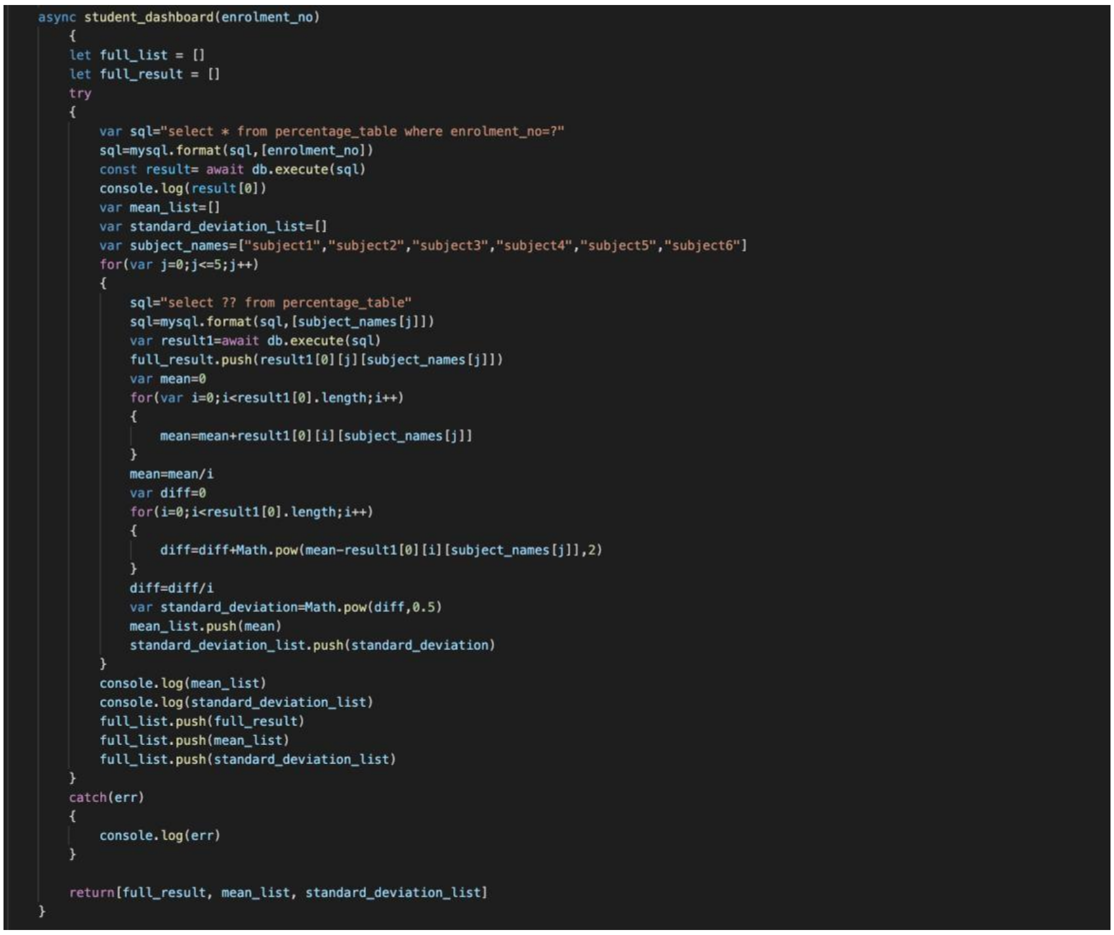
Copyright © Automated Attendance Management System 2021.

Fig 7.6 : Student Dashboard



```
// STUDENT DASHBOARD
app.post("/studentdashboard",async function (req,res){
    var [x,y,z] = await obj.student_dashboard(req.body.rno)

    console.log(x)
    res.status(200).json({
        "percent":x,
        "mean":y,
        "stddev":z,
        "error":"none"
    })
});
```

Once the student is authenticated the following dashboard loads.



```
async student_dashboard(enrolment_no)
    {
    let full_list = []
    let full_result = []
    try
    {
        var sql="select * from percentage_table where enrolment_no=?"
        sql=mysql.format(sql,[enrolment_no])
        const result= await db.execute(sql)
        console.log(result[0])
        var mean_list=[]
        var standard_deviation_list=[]
        var subject_names=["subject1","subject2","subject3","subject4","subject5","subject6"]
        for(var j=0;j<=5;j++)
        {
            sql="select ?? from percentage_table"
            sql=mysql.format(sql,[subject_names[j]])
            var result1=await db.execute(sql)
            full_result.push(result1[0][j][subject_names[j]])
            var mean=0
            for(var i=0;i<result1[0].length;i++)
            {
                mean=mean+result1[0][i][subject_names[j]]
            }
            mean=mean/i
            var diff=0
            for(i=0;i<result1[0].length;i++)
            {
                diff=diff+Math.pow(mean-result1[0][i][subject_names[j]],2)
            }
            diff=diff/i
            var standard_deviation=Math.pow(diff,0.5)
            mean_list.push(mean)
            standard_deviation_list.push(standard_deviation)
        }
        console.log(mean_list)
        console.log(standard_deviation_list)
        full_list.push(full_result)
        full_list.push(mean_list)
        full_list.push(standard_deviation_list)
    }
    catch(err)
    {
        console.log(err)
    }

    return[full_result, mean_list, standard_deviation_list]
}
```

This is how the calculations for all the information shown in the student dashboard is computed in the backend using the data.

7.3 Teacher Application

Teachers will have only one dashboard where the can view the attendance of the students in their class. They are not given a signup process as they will need an admin to create an account for them.

The teacher login page is accessible from the landing page.

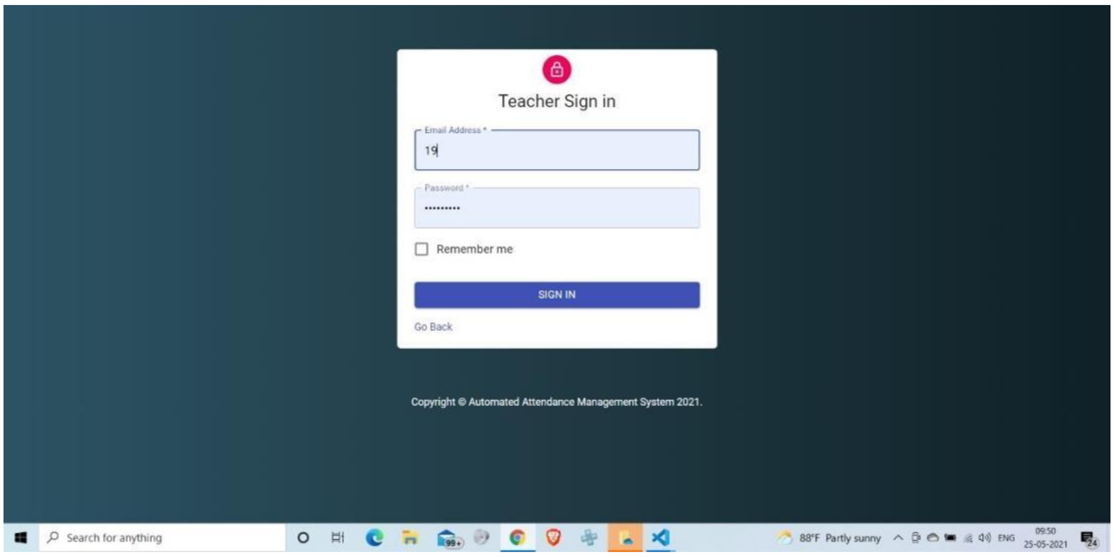
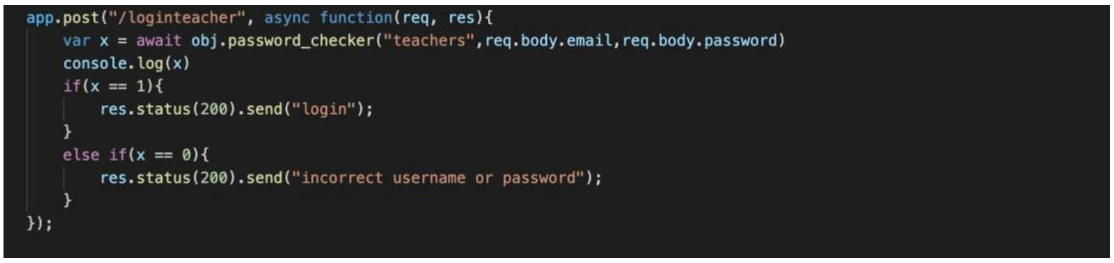


Fig 7.7 : Teacher Sign in

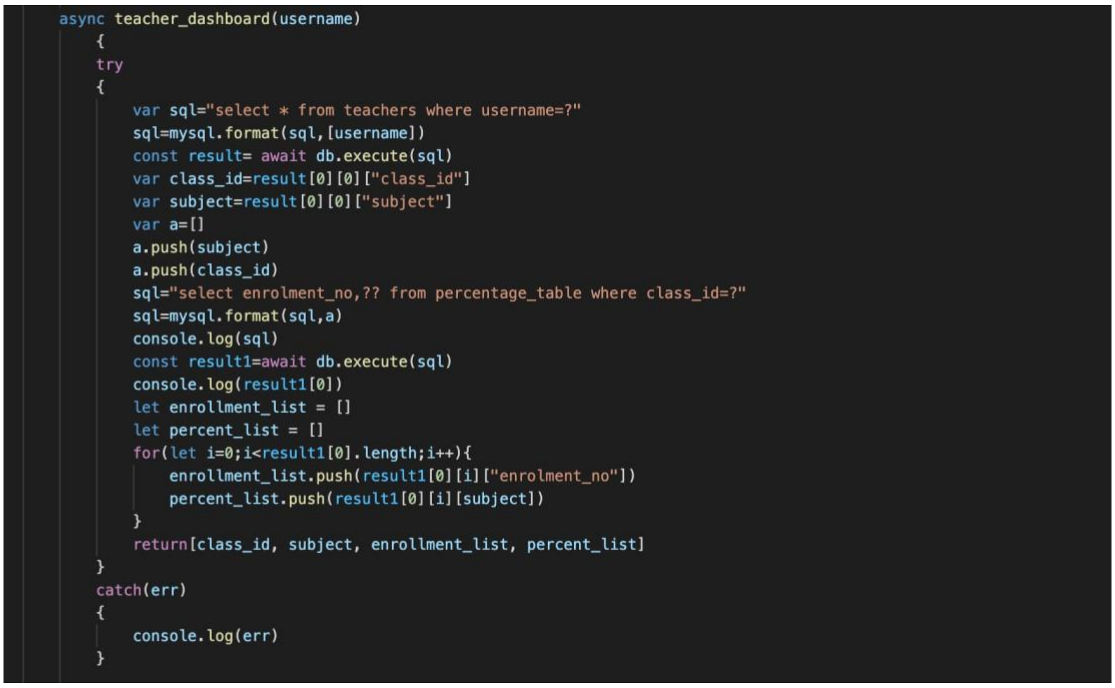
The login credentials are passed on to the SQL server which then authenticates it and returns if authenticated or not, it follows the same logic from the student authentication module.



```
app.post("/loginteacher", async function(req, res){
    var x = await obj.password_checker("teachers",req.body.email,req.body.password)
    console.log(x)
    if(x == 1){
        res.status(200).send("login");
    }
    else if(x == 0){
        res.status(200).send("incorrect username or password");
    }
});
```

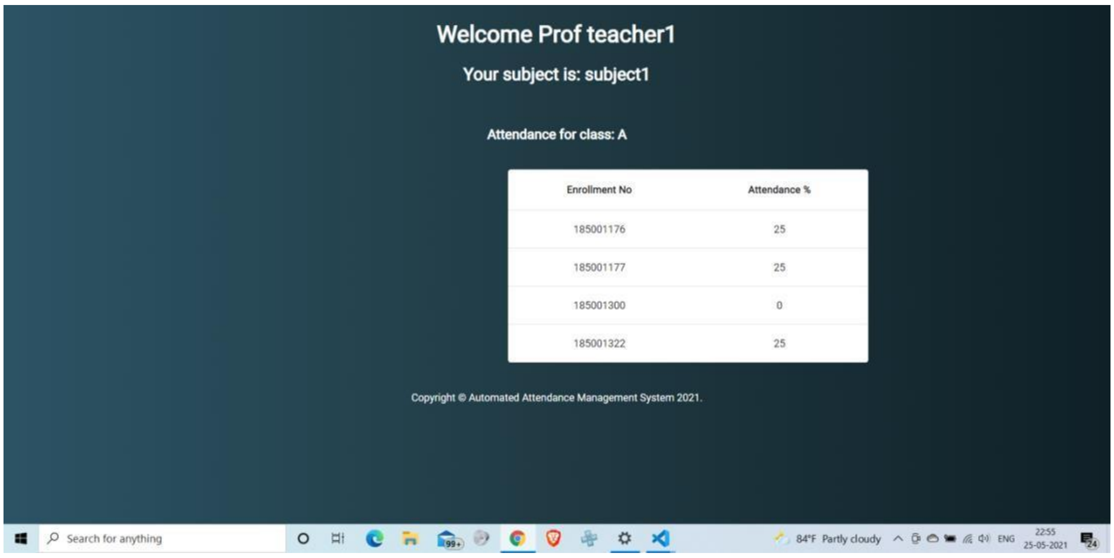
Once the login is authenticated with the database, the teacher is taken to a dashboard where they can view the attendance of all the students present in their class.

The dashboard is computed in the backend using the following logic.



```
async teacher_dashboard(username)
    {
    try
    {
        var sql="select * from teachers where username=?"
        sql=mysql.format(sql,[username])
        const result= await db.execute(sql)
        var class_id=result[0][0]["class_id"]
        var subject=result[0][0]["subject"]
        var a=[]
        a.push(subject)
        a.push(class_id)
        sql="select enrolment_no,?? from percentage_table where class_id=?"
        sql=mysql.format(sql,a)
        console.log(sql)
        const result1=await db.execute(sql)
        console.log(result1[0])
        let enrollment_list = []
        let percent_list = []
        for(let i=0;i<result1[0].length;i++){
            enrollment_list.push(result1[0][i]["enrolment_no"])
            percent_list.push(result1[0][i][subject])
        }
        return[class_id, subject, enrollment_list, percent_list]
    }
    catch(err)
    {
        console.log(err)
    }
```

The final teacher dashboard when the teacher completes the login flow and the data is fetched from the MySQL database:



Welcome Prof teacher1

Your subject is: subject1

Attendance for class: A

Enrollment No	Attendance %
185001176	25
185001177	25
185001300	0
185001322	25

Copyright © Automated Attendance Management System 2021.

Fig 7.8 : Teacher Dashboard

7.4 Admin Application

Admin can login using the credentials given to him, the login page can be accessed from the landing page.

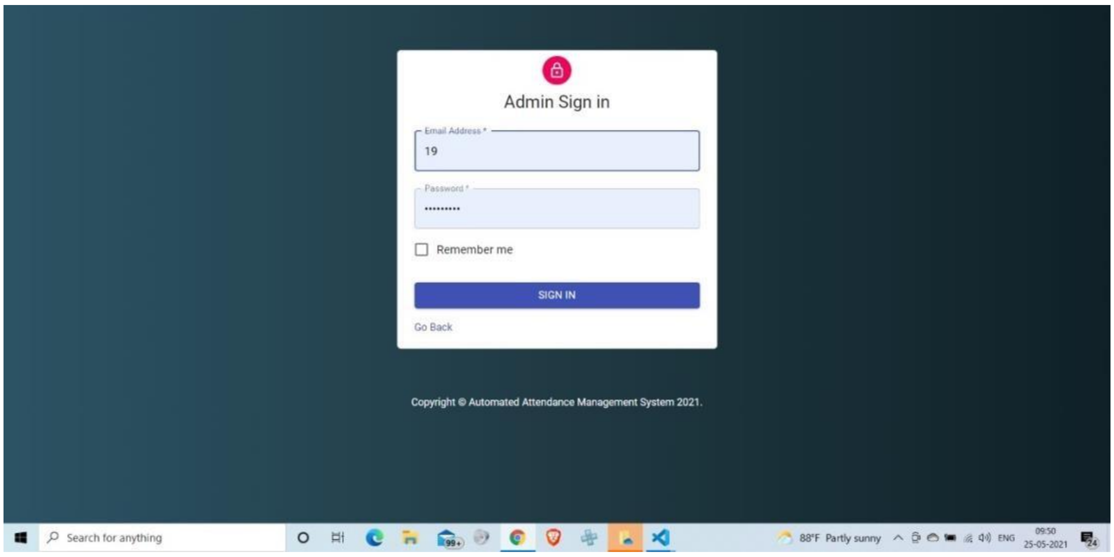
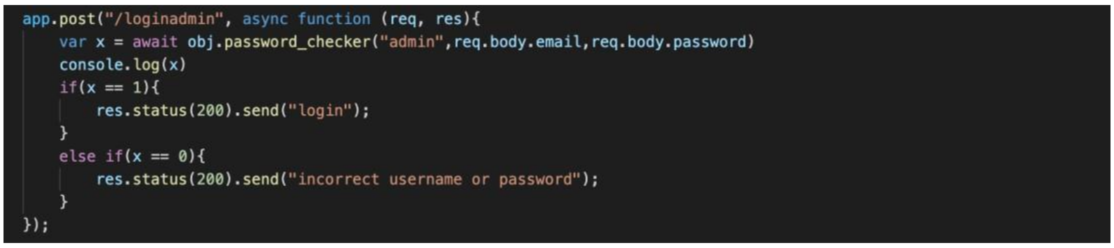


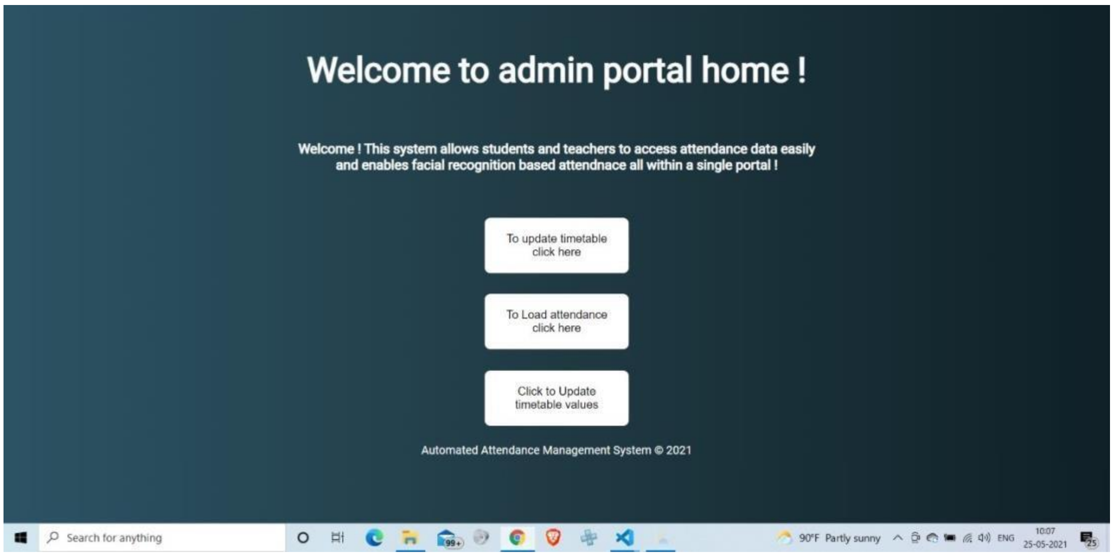
Fig 7.9 : Admin Sign in

The authentication for the admin account is same as mentioned in the student authentication flow



```
app.post("/loginadmin", async function (req, res){
    var x = await obj.password_checker("admin",req.body.email,req.body.password)
    console.log(x)
    if(x == 1){
        res.status(200).send("login");
    }
    else if(x == 0){
        res.status(200).send("incorrect username or password");
    }
});
```

Once log in is authenticated, the admin is taken to a landing page where he is allowed to perform a few changes from the backend in the database.


Welcome to admin portal home !

Welcome ! This system allows students and teachers to access attendance data easily and enables facial recognition based attendnace all within a single portal !

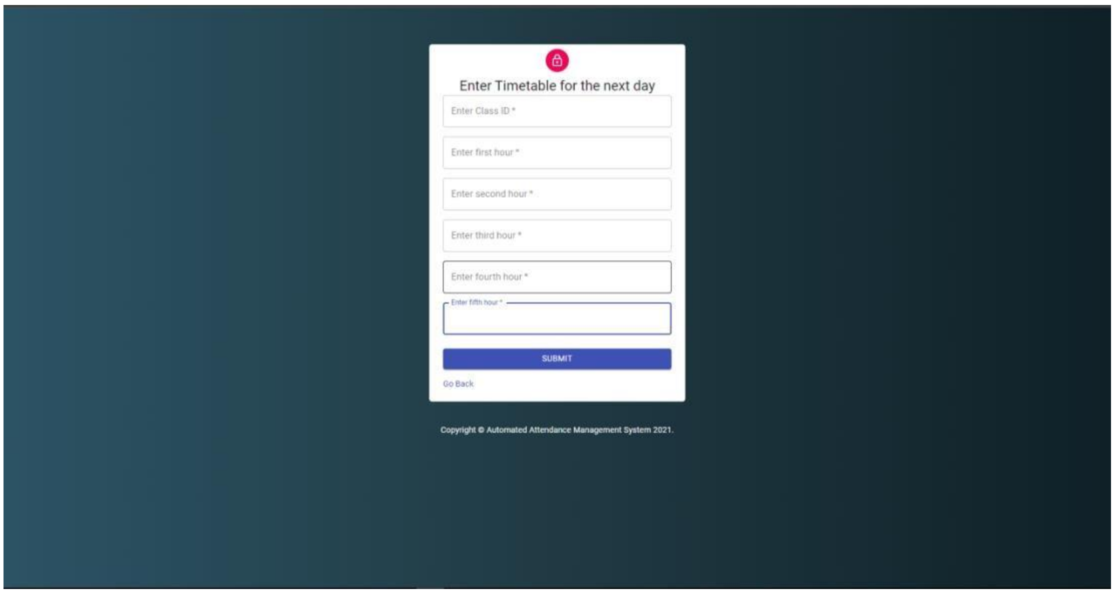
Automated Attendance Management System © 2021

To update timetable click here
To Load attendance click here
Click to Update timetable values
Search for anything
90°F Partly sunny
ENG
10:07
25-05-2021

Fig 7.10 : Admin Dashboard

7.4.1 Change the timetable

The admin has the option to change only time table of a particular day in case of any changes on. A daily basis.


Enter Timetable for the next day

Enter Class ID *

Enter first hour *

Enter second hour *

Enter third hour *

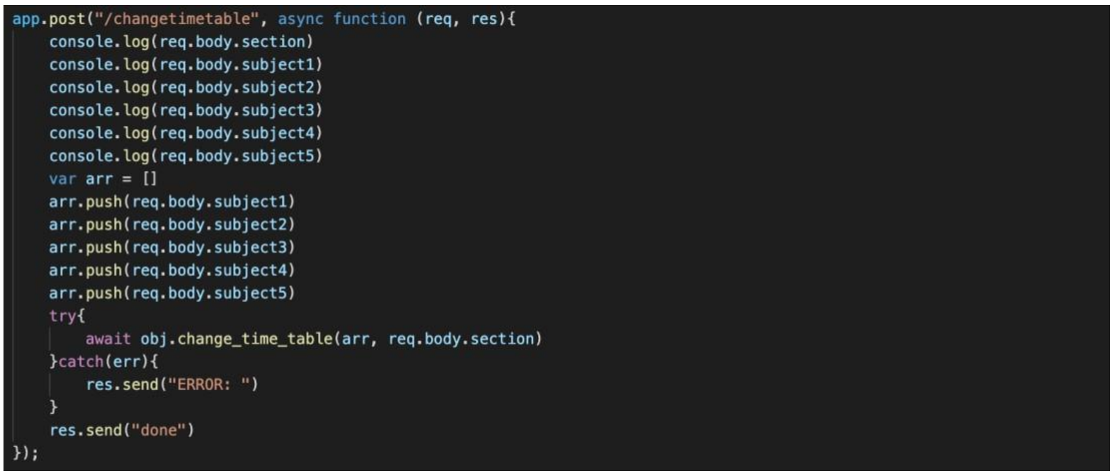
Enter fourth hour *

Enter fifth hour *

Copyright © Automated Attendance Management System 2021.

Fig 7.11 : Change the timetable

On clicking choosing the timetable order and clicking submit, the order inputted by the admin is passed as an array into the MySQL database.



```
app.post("/changetimetable", async function (req, res){
    console.log(req.body.section)
    console.log(req.body.subject1)
    console.log(req.body.subject2)
    console.log(req.body.subject3)
    console.log(req.body.subject4)
    console.log(req.body.subject5)
    var arr = []
    arr.push(req.body.subject1)
    arr.push(req.body.subject2)
    arr.push(req.body.subject3)
    arr.push(req.body.subject4)
    arr.push(req.body.subject5)
    try{
        await obj.change_time_table(arr, req.body.section)
    }catch(err){
        res.send("ERROR: ")
    }
    res.send("done")
});
```

7.4.2 Load Attendance

The Admin can upload all the images the faculty captures into the DB to award the attendance present to people in the image using the facial recognition.

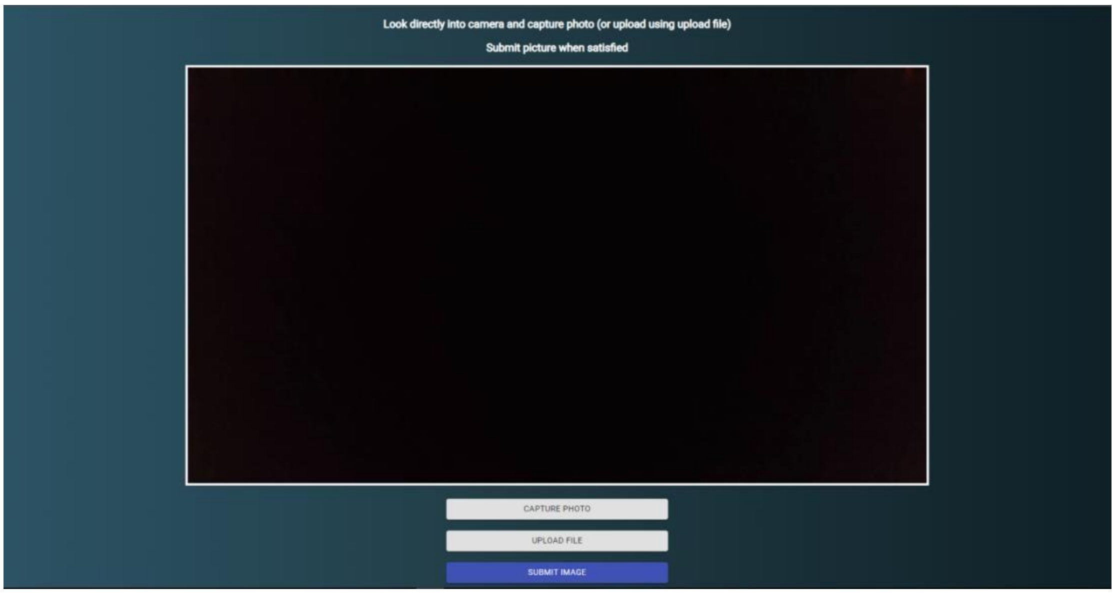

Look directly into camera and capture photo (or upload using upload file)
Submit picture when satisfied
CAPTURE PHOTO
UPLOAD FILE
SUBMIT IMAGE

Fig 7.12 : Timetable Load

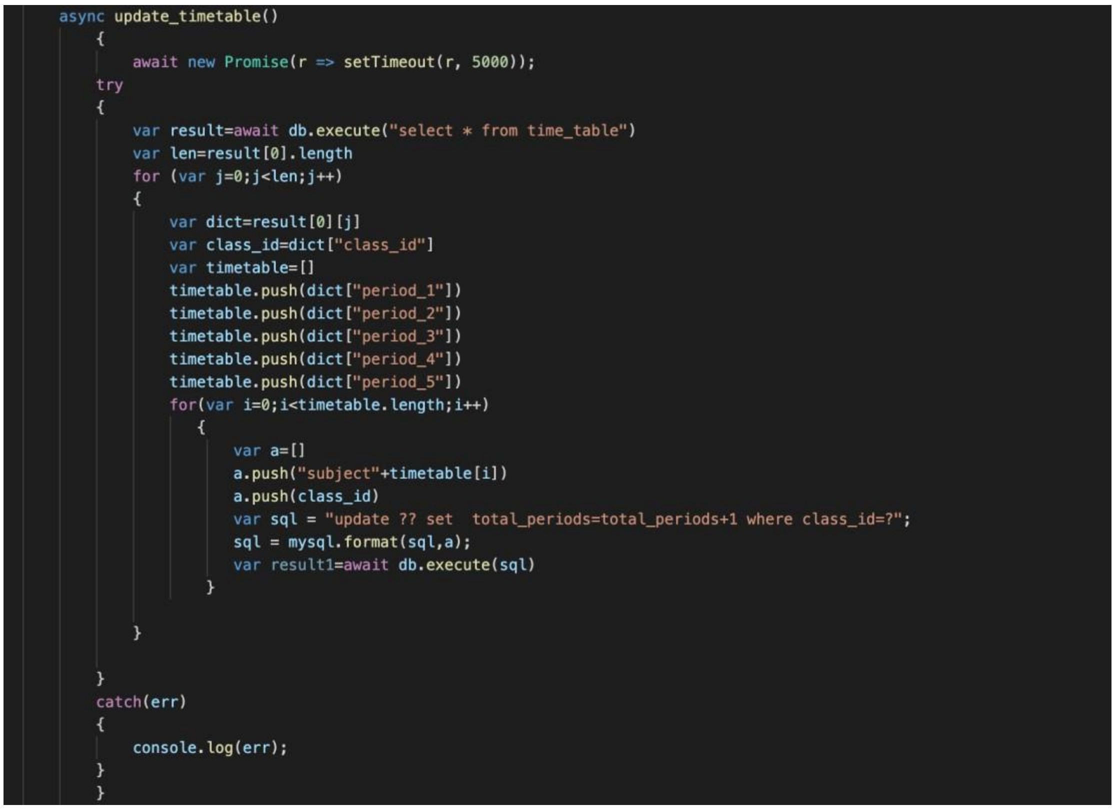
The admin has the option to capture an image or upload an image captured by the faculties and upload it into the database to award attendance to

The facial recognition part will be explained in detail in the next chapter.

7.4.3 Update Timetable

The admin has to manually update the classes for everyone on a daily basis to prevent false calculations, he will need to confirm the order once after checking the order and click on the update time table button to add the classes attended in the students time table.

In the backend, it'll add +1 to all the students present in the class to the total periods which is used in attendance calculation.



```
async update_timetable()
    {
        await new Promise(r => setTimeout(r, 5000));
    try
    {
        var result=await db.execute("select * from time_table")
        var len=result[0].length
        for (var j=0;j<len;j++)
        {
            var dict=result[0][j]
            var class_id=dict["class_id"]
            var timetable=[]
            timetable.push(dict["period_1"])
            timetable.push(dict["period_2"])
            timetable.push(dict["period_3"])
            timetable.push(dict["period_4"])
            timetable.push(dict["period_5"])
            for(var i=0;i<timetable.length;i++)
              {
                var a=[]
                a.push("subject"+timetable[i])
                a.push(class_id)
                var sql = "update ?? set  total_periods=total_periods+1 where class_id=?";
                sql = mysql.format(sql,a);
                var result1=await db.execute(sql)
              }

        }

    }
    catch(err)
    {
        console.log(err);
    }
    }
```

Chapter 8

FACE RECOGNITION

8.1 Face Detection models

8.1.1 SSD Mobilenet V1

This project uses a Single Shot Multibox Detector (SSD) based on Mobilenet V1. The neural net computes the locations of each face in an image and returns the bounding boxes along with its probability for each face. It is aimed to obtain the highest accuracy in detecting the bounding boxes in the lowest inference time. The size of the Quantized model is about 5.4 MB.

8.1.2 Tiny Face Detector

The Tiny Face Detector is a much faster ,smaller and much more lesser resource consuming compared to the SSD Mobilenet V1, however it performs slightly worse in detecting smaller faces. The model runs better on mobile devices and resource limited clients, and hence was used.

The size of the quantized model is only 190 KB. The Face Detector is trained on a custom dataset of ~14K images which are labelled with bounding boxes. The model is also trained to predict bounding boxes which cover the entire facial feature points. And hence produces better results in combination with subsequent face landmark detection.

The model is an even tinner version of Tiny Yolo V2.

8.1.3 68 Point Face Landmark Detection Models

This package implements a very accurate, lightweight and fast 68 point face landmark detector. The default model is 350 KB and the tiny model is only 80 KB, Both models employ the ideas of depth wise separable convolutions as well as densely connected blocks. The models have been trained on a dataset of ~35k face images labelled with 68 face landmark points.

8.1.4 Face Recognition model

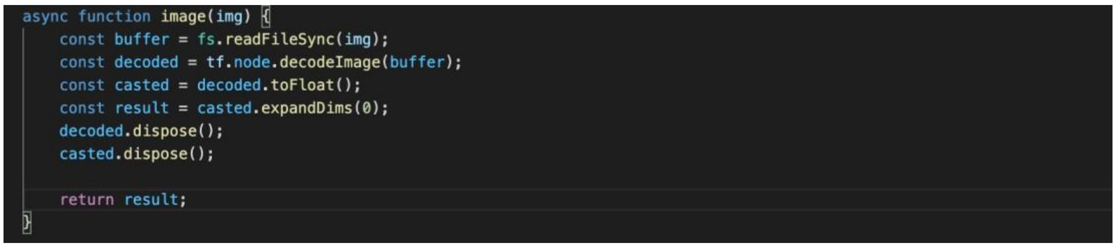
For face recognition, a ResNet-34 like architecture is implemented to compute a face descriptor (a feature vector with 128 values) from any given face image, which is used to describe the characteristics of a person’s face. You can determine the similarity of two arbitrary faces by comparing their face descriptors, for example by computing the Euclidean distance or using any other classifier of your choice.
The size of the quantized model is roughly 6.2 MB

8.1.5 Face Expression Recognition Model

The face expression recognition model is fast, lightweight and provides reasonable accuracy. The model has a size of roughly 310kb and it employs depth wise separable convolutions and densely connected blocks. It has been trained on a variety of images from publicly available datasets as well as images scraped from the web. Note, that wearing glasses might decrease the accuracy of the prediction results.

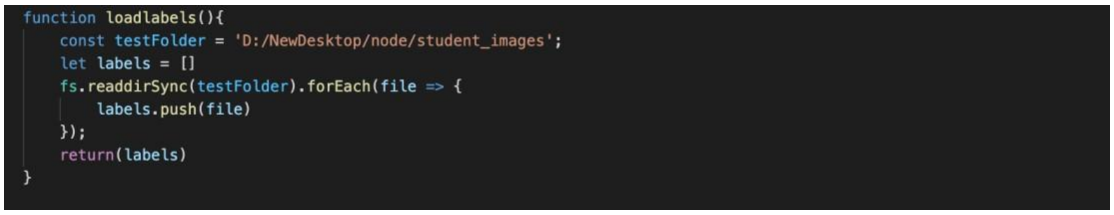
8.2 Implementation

When the image is loaded on the admin panel the image, the image is first converted into the required tensor format.



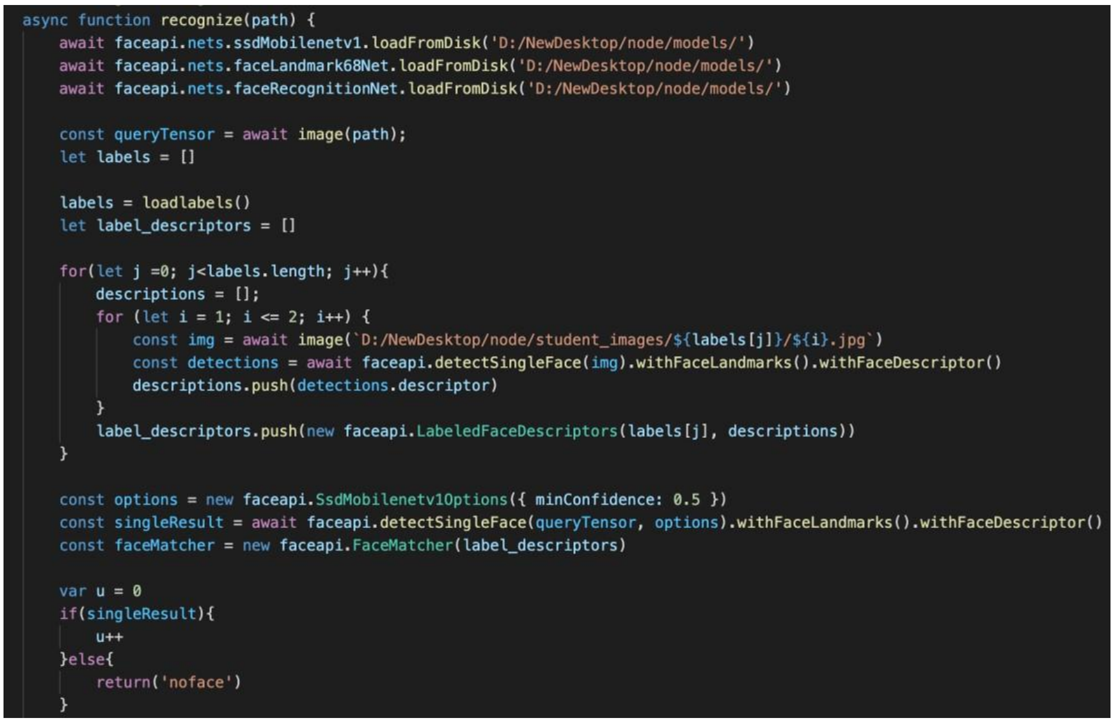
```
async function image(img) {
    const buffer = fs.readFileSync(img);
    const decoded = tf.node.decodeImage(buffer);
    const casted = decoded.toFloat();
    const result = casted.expandDims(0);
    decoded.dispose();
    casted.dispose();

    return result;
}
```

The student labels are loaded from the directory.



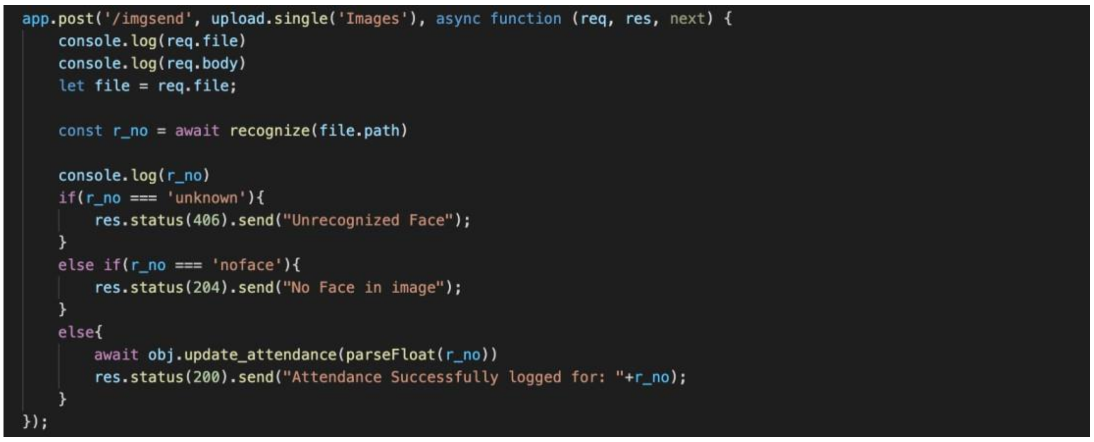
```
function loadlabels(){
    const testFolder = 'D:/NewDesktop/node/student_images';
    let labels = []
    fs.readdirSync(testFolder).forEach(file => {
        labels.push(file)
    });
    return(labels)
}
```

The models mentioned in the previous sections is used to first find out the number of faces in the image and locate them and then the find out the closest match to these faces by converting them to string and comparing it to the faces stored in the database by students when signing up for the platform.



```javascript
async function recognize(path) {
    await faceapi.nets.ssdMobilenetv1.loadFromDisk('D:/NewDesktop/node/models/')
    await faceapi.nets.faceLandmark68Net.loadFromDisk('D:/NewDesktop/node/models/')
    await faceapi.nets.faceRecognitionNet.loadFromDisk('D:/NewDesktop/node/models/')

    const queryTensor = await image(path);
    let labels = []

    labels = loadlabels()
    let label_descriptors = []

    for(let j =0; j<labels.length; j++){
        descriptions = [];
        for (let i = 1; i <= 2; i++) {
            const img = await image(`D:/NewDesktop/node/student_images/${labels[j]}/${i}.jpg`)
            const detections = await faceapi.detectSingleFace(img).withFaceLandmarks().withFaceDescriptor()
            descriptions.push(detections.descriptor)
        }
        label_descriptors.push(new faceapi.LabeledFaceDescriptors(labels[j], descriptions))
    }

    const options = new faceapi.SsdMobilenetv1Options({ minConfidence: 0.5 })
    const singleResult = await faceapi.detectSingleFace(queryTensor, options).withFaceLandmarks().withFaceDescriptor()
    const faceMatcher = new faceapi.FaceMatcher(label_descriptors)

    var u = 0
    if(singleResult){
        u++
    }else{
        return('noface')
    }
```

Then the face recognition is performed and the attendance is awarded to the candidates



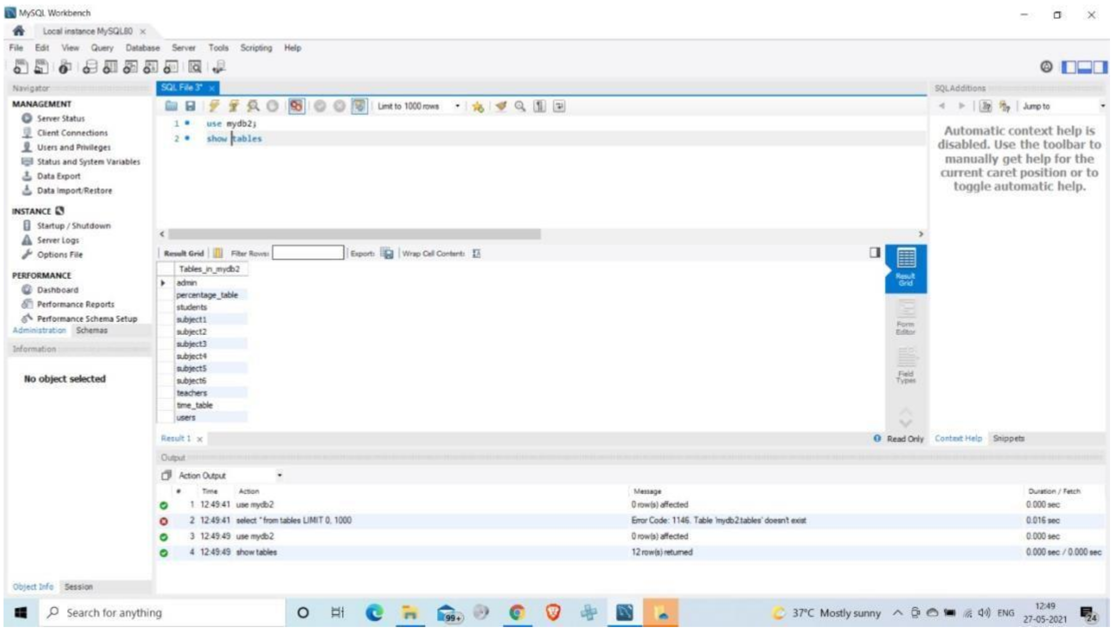
```javascript
app.post('/imgsend', upload.single('Images'), async function (req, res, next) {
    console.log(req.file)
    console.log(req.body)
    let file = req.file;

    const r_no = await recognize(file.path)

    console.log(r_no)
    if(r_no === 'unknown'){
        res.status(406).send("Unrecognized Face");
    }
    else if(r_no === 'noface'){
        res.status(204).send("No Face in image");
    }
    else{
        await obj.update_attendance(parseFloat(r_no))
        res.status(200).send("Attendance Successfully logged for: "+r_no);
    }
});
```

Chapter 9

DATABASE

9.1 Implementation

List of tables present in the database for the system to operate



```
use mydb2;
show tables
```

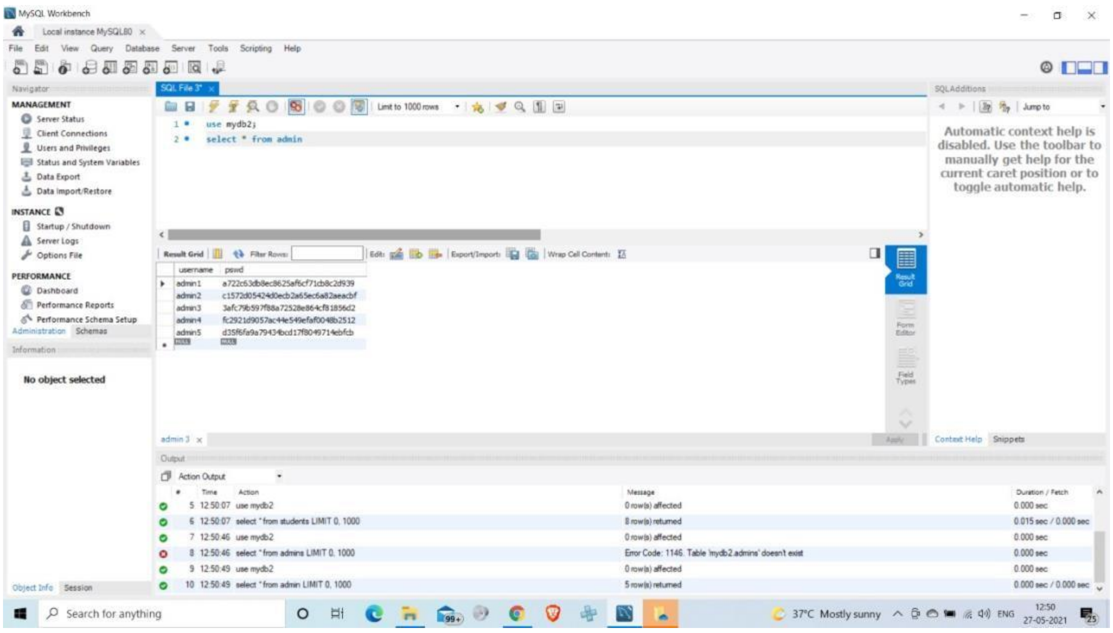
Tables_in_mydb2
admin
percentage_table
students
subject1
subject2
subject3
subject4
subject5
subject6
teachers
time_table
users

#	Time	Action	Message	Duration / Fetch
1	12:49:41	use mydb2	0 row(s) affected	0.000 sec
2	12:49:41	select * from tables LIMIT 0, 1000	Error Code: 1146. Table 'mydb2.tables' doesn't exist	0.016 sec
3	12:49:49	use mydb2	0 row(s) affected	0.000 sec
4	12:49:49	show tables	12 row(s) returned	0.000 sec / 0.000 sec

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Fig 9.1 : Tables

Admin table with login details and encrypted password



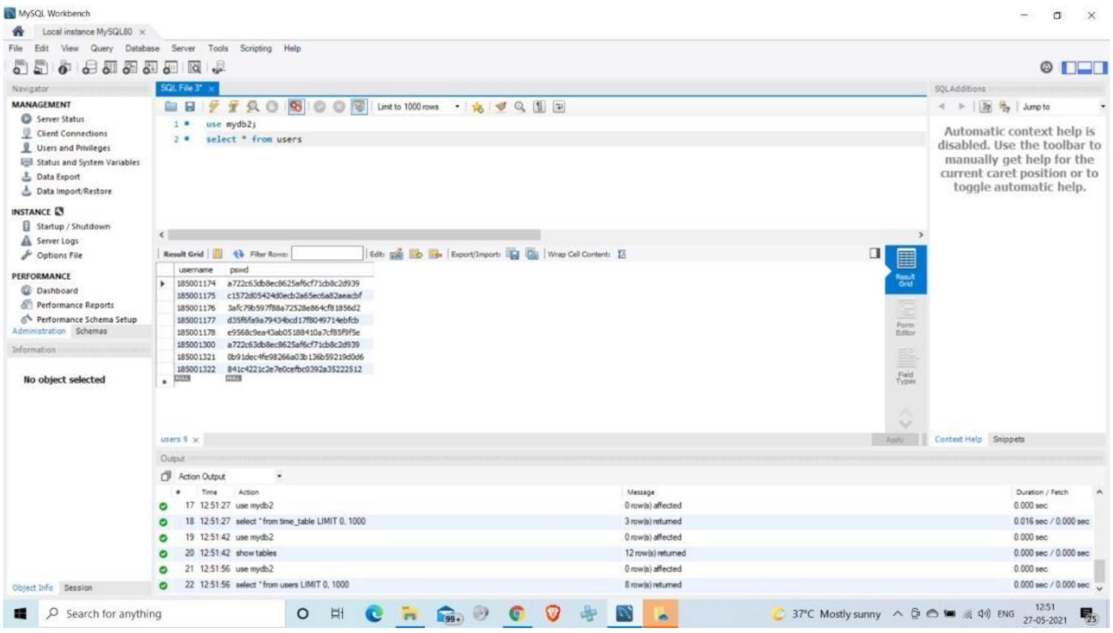
```
use mydb2;
select * from admin
```

username	pswd
admin1	a722c63db8ec8625af6cf71cb8c2d939
admin2	c1572d05424d0ecb2a65ec6a82aeacbf
admin3	3afc79b597f88a72528e864cf81856d2
admin4	fc2921d9057ac44e549efaf0048b2512
admin5	d35f6fa9a79434bcd17f8049714ebfcb
NULL	NULL

```
#   Time      Action                                     Message                                           Duration / Fetch
5   12:50:07  use mydb2                                  0 row(s) affected                                 0.000 sec
6   12:50:07  select * from students LIMIT 0, 1000       8 row(s) returned                                 0.015 sec / 0.000 sec
7   12:50:46  use mydb2                                  0 row(s) affected                                 0.000 sec
8   12:50:46  select * from admins LIMIT 0, 1000         Error Code: 1146. Table 'mydb2.admins' doesn't exist  0.000 sec
9   12:50:49  use mydb2                                  0 row(s) affected                                 0.000 sec
10  12:50:49  select * from admin LIMIT 0, 1000          5 row(s) returned                                 0.000 sec / 0.000 sec
```

Fig 9.2 : Admin Database

The teacher and student tables are also have the same schema where the teacher/student user name is stored and the password is stored after encrypting using a MD5 hash.

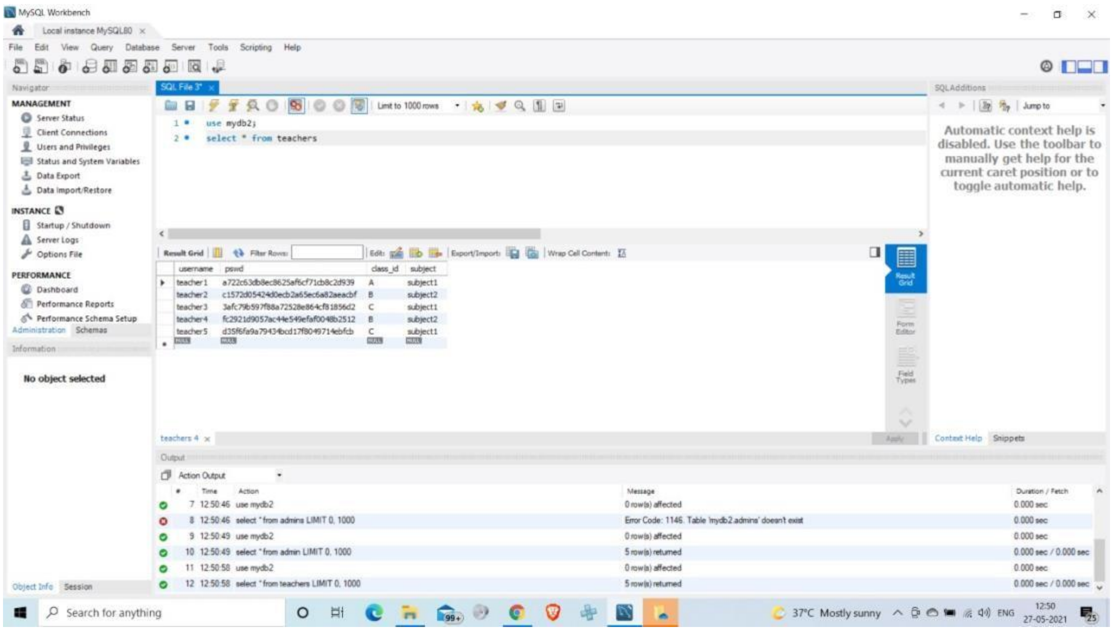


```
use mydb2;
select * from users
```

username	pswd
185001174	a722c63db8ec8625af6cf71cb8c2d939
185001175	c1572d05424d0ecb2a65ec6a82aeacbf
185001176	3afc79b597f88a72528e864cf81856d2
185001177	d35f6fa9a79434bcd17f8049714ebfcb
185001178	e9568c9ea43ab05188410a7cf85f9f5e
185001300	a722c63db8ec8625af6cf71cb8c2d939
185001321	0b91dec4fe98266a03b136b59219d0d6
185001322	841c4221c2e7e0cefbc0392a35222512
NULL	NULL

```
#   Time      Action                                     Message               Duration / Fetch
17  12:51:27  use mydb2                                  0 row(s) affected     0.000 sec
18  12:51:27  select * from time_table LIMIT 0, 1000     3 row(s) returned     0.016 sec / 0.000 sec
19  12:51:42  use mydb2                                  0 row(s) affected     0.000 sec
20  12:51:42  show tables                                12 row(s) returned    0.000 sec / 0.000 sec
21  12:51:56  use mydb2                                  0 row(s) affected     0.000 sec
22  12:51:56  select * from users LIMIT 0, 1000          8 row(s) returned     0.000 sec / 0.000 sec
```

Fig 9.3 : Student Database



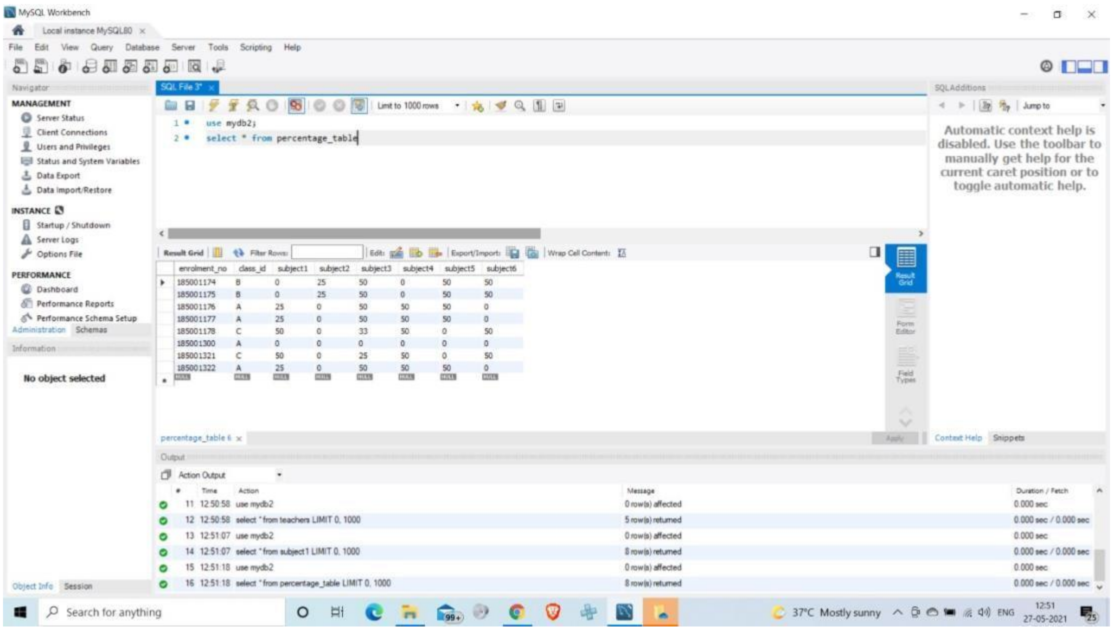
```
use mydb2;
select * from teachers
```

username	pswd	class_id	subject
teacher1	a722c63db8ec8625af6cf71db8c2d939	A	subject1
teacher2	c1572d05424d0ecb2a65ec6a82aeacbf	B	subject2
teacher3	3afc79b597f88a72528e864cf81856d2	C	subject1
teacher4	fc2921d9057ac44e549efaf0048b2512	B	subject2
teacher5	d35f6fa9a79434bcd17f8049714ebfcb	C	subject1
NULL	NULL	NULL	NULL

```
#   Time      Action                                   Message                                            Duration / Fetch
7   12:50:46  use mydb2                                0 row(s) affected                                  0.000 sec
8   12:50:46  select * from admins LIMIT 0, 1000       Error Code: 1146. Table 'mydb2.admins' doesn't exist  0.000 sec
9   12:50:49  use mydb2                                0 row(s) affected                                  0.000 sec
10  12:50:49  select * from admin LIMIT 0, 1000        5 row(s) returned                                  0.000 sec / 0.000 sec
11  12:50:58  use mydb2                                0 row(s) affected                                  0.000 sec
12  12:50:58  select * from teachers LIMIT 0, 1000     5 row(s) returned                                  0.000 sec / 0.000 sec
```

Fig 9.4 : Teacher Database

The attendance percentage of the students is stored in a table which returns the computed value after attendance is updated by the admin every day.



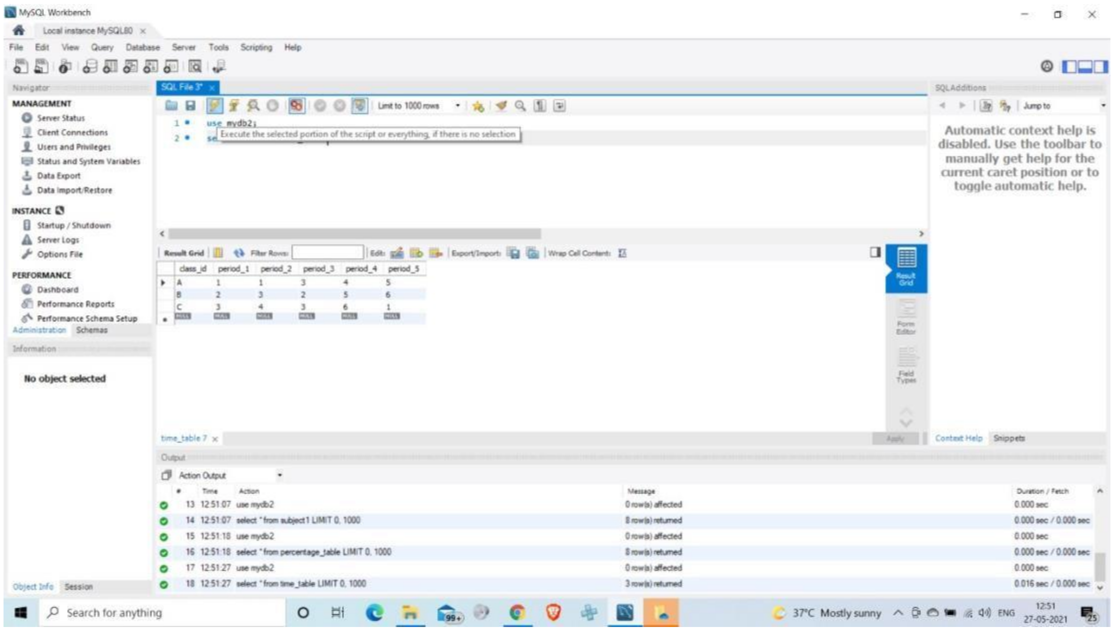
```
use mydb2;
select * from percentage_table
```

enrolment_no	class_id	subject1	subject2	subject3	subject4	subject5	subject6
185001174	B	0	25	50	0	50	50
185001175	B	0	25	50	0	50	50
185001176	A	25	0	50	50	50	0
185001177	A	25	0	50	50	50	0
185001178	C	50	0	33	50	0	50
185001300	A	0	0	0	0	0	0
185001321	C	50	0	25	50	0	50
185001322	A	25	0	50	50	50	0
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
#   Time      Action                                         Message             Duration / Fetch
11  12:50:58  use mydb2                                      0 row(s) affected   0.000 sec
12  12:50:58  select * from teachers LIMIT 0, 1000           5 row(s) returned   0.000 sec / 0.000 sec
13  12:51:07  use mydb2                                      0 row(s) affected   0.000 sec
14  12:51:07  select * from subject1 LIMIT 0, 1000           8 row(s) returned   0.000 sec / 0.000 sec
15  12:51:18  use mydb2                                      0 row(s) affected   0.000 sec
16  12:51:18  select * from percentage_table LIMIT 0, 1000   8 row(s) returned   0.000 sec / 0.000 sec
```

Fig 9.5 : Attendance percentage

The Time table is stored in the database in the following way by using the subject id’s.



```
use mydb2;
```

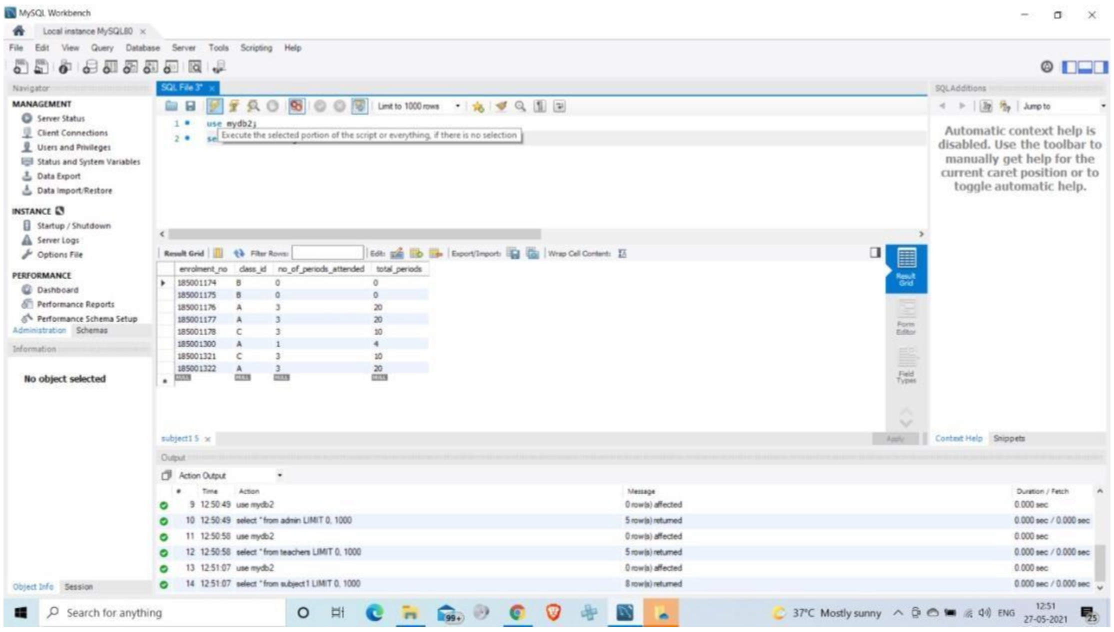
Execute the selected portion of the script or everything, if there is no selection

class_id	period_1	period_2	period_3	period_4	period_5
A	1	1	3	4	5
B	2	3	2	5	6
C	3	4	3	6	1
NULL	NULL	NULL	NULL	NULL	NULL

```
#   Time      Action                                          Message             Duration / Fetch
13  12:51:07  use mydb2                                       0 row(s) affected   0.000 sec
14  12:51:07  select * from subject1 LIMIT 0, 1000            8 row(s) returned   0.000 sec / 0.000 sec
15  12:51:18  use mydb2                                       0 row(s) affected   0.000 sec
16  12:51:18  select * from percentage_table LIMIT 0, 1000    8 row(s) returned   0.000 sec / 0.000 sec
17  12:51:27  use mydb2                                       0 row(s) affected   0.000 sec
18  12:51:27  select * from time_table LIMIT 0, 1000          3 row(s) returned   0.016 sec / 0.000 sec
```

Fig 9.6 : Time Table

The attendance for each subject along with the details of all the student attending that subject, the class they belong to and the classes they attended along with the total number of classes conducted is stored in this format:



```
use mydb2;
```

Execute the selected portion of the script or everything, if there is no selection

enrolment_no	class_id	no_of_periods_attended	total_periods
185001174	B	0	0
185001175	B	0	0
185001176	A	3	20
185001177	A	3	20
185001178	C	3	10
185001300	A	1	4
185001321	C	3	10
185001322	A	3	20
NULL	NULL	NULL	NULL

```
#   Time      Action                                   Message             Duration / Fetch
9   12:50:49  use mydb2                                0 row(s) affected   0.000 sec
10  12:50:49  select * from admin LIMIT 0, 1000        5 row(s) returned   0.000 sec / 0.000 sec
11  12:50:58  use mydb2                                0 row(s) affected   0.000 sec
12  12:50:58  select * from teachers LIMIT 0, 1000     5 row(s) returned   0.000 sec / 0.000 sec
13  12:51:07  use mydb2                                0 row(s) affected   0.000 sec
14  12:51:07  select * from subject1 LIMIT 0, 1000     8 row(s) returned   0.000 sec / 0.000 sec
```

Fig 9.7 : Subject Attendance

Chapter 10

CONCLUSION

This is an attempt to ease the attendance process to ensure time is not wasted for something that is as trivial as attendance and the entire process is made sure to ensure the whole attendance is take in less than a minute and is very easy for the faculty to complete.

The most important outcome was the complete attendance process which used to take a significant amount of time during the lecture hours has been reduced drastically. This system can be accessed from anywhere remotely which is an advantage as the students will be able to check their attendance remotely without the need to be present on the same network.

The system is also very economical to setup, it can be run on any PC and the input should be an image provided which can be captured by using any imaging equipment. It can also be very scalable as the only things needed to be upgraded would be storage and compute power which will be even more easier to upgrade if the entire platform is hosted on the cloud.

The most important outcome was learning the process of building an entire app from scratch end-end and keeping track of various functionalities I wanted to integrate and figuring out how other apps in the market has implemented similar functionalities and trying to integrate the same. There was also a learning curve on integrating the various modules built to together as well as a lot of research when something went wrong while integrating the modules. It was an extremely good experience on building the application and the platform end-end which gave me a lot of insight into designing, developing testing, debugging and documenting all of the work I have done. Another most important learning outcome which I realized while creating this platform is the importance of planning and the usage of PRD’s while creating the product. With that being said there is scope for improvement in the product to make it even more easier to use and understand which I will be explaining in the next section.

Chapter 11

Future Works

The project although helps in making the process easier and efficient, there are a few areas which could be made to make it even more user friendly.

The student dashboard consists of the various information which showcases the students attendance in a table, it showcases raw data computed in the backend.
It would be more easier to understand the data if it was represented graphically using some different types of indicators such as charts, graphs, etc.

sometimes the student might be present in the blind spot in an image where the algorithm might not be able to calculate the attendance, in these cases he might not be awarded any attendance, in these cases the student might need to go to the admin and request for the attendance. To make this process easier, it would be better to have a ticketing process where a student can appeal if he is not awarded attendance or wrongly awarded one, then the system will show him/her the raw image from which the attendance was computed where they can mark themselves in the image which might help the admin to award them the attendance in case of a mistake.

The teachers dashboard is very basic right now and displays the list of all the students present in their class along with their attendance percentages, it can include some more features such as computing the attendance which present in the admin panel as it would mean the faculty could complete the entire process. It can also be modified a bit to show all the details of the students present in the class rather than only the roll number and attendance.

The entire model relies on the image being given to perform the function on is of a high quality, however there has been many advancements such as use of biometric technologies to compute attendance which however costs higher to install and maintain. We can also explore other methods to implement in this platform with the use of advancement in technology as NFC, dynamic QR generators, etc.

Chapter 12

REFERENCES

Luca Bertinetto, João F. Henriques, Jack Valmadre, Philip H. S. Torr, Andrea Vedaldi (June 2016). Learning feed-forward one-shot learners. arXiv.org. https://arxiv.org/abs/1606.05233v1

Sukalpa Chanda, Asish Chakrapani GV, Anders Brun, Anders Hast, Umapada Pal and David Doermann (2019), "Face Recognition - A One-Shot Learning Perspective" i n 2019 15th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)

P. Jonathon Phillips , Patrick J. Flynn , Todd Scruggs , Kevin W. Bowyer, Jin Chang , Kevin Hoffman , Joe Marques , Jaesik Min , William Worek (2005), "Overview of the Face Recognition Grand Challenge" in I EEE Conference on Computer Vision and Pattern Recognition

Florian Schroff, Dmitry Kalenichenko, James Philbin (2015), "FaceNet: A Unified Embedding for Face Recognition and Clustering" in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. arXiv.org. h ttps://arxiv.org/abs/1503.03832v3

Kairos: Serving Businesses with Face Recognition. (n.d.). Retrieved May 16, 2020, from https://www.kairos.com/

Lodha, R., Gupta, S., Jain, H., & Narula, H. (2015). Bluetooth Smart based attendance management system. Procedia Computer Science, 45(C), 524–527. https://doi.org/10.1016/j.procs.2015.03.094

Zhang, K., Zhang, Z., Li, Z., & Qiao, Y. (2016). Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. IEEE Signal Processing Letters, 23(10), 1499–1503. https://doi.org/10.1109/LSP.2016.2603342

Zoph, B., Vasudevan, V., Shlens, J., & Le, Q. V. (2018). Learning Transferable Architectures for Scalable Image Recognition. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 8697–8710. https://doi.org/10.1109/CVPR.2018.00907