National University of Computer and Emerging Sciences

Project Title: **Facial Attendee**

FACIAL ATTENDEE

Easiness in taking Attendence

Submitted by:

Mehdi Raza Rajani - 16k3904

Mohammad Raza Vasnani 16k3890

Mohammad Ammar Rizwan 16k3862

Tahir Raza Hemani 16k3905

(Students of Section - F)

May 22, 2019

Letter of Transmittal

Dr. Nauman Ghazi
Assistant Professor
NUCES, FAST Karachi
St-4 Sector 17-D On National Highway Karachi , Pakistan

Sub: Letter of Transmittal

Dear

It is indeed a great pleasure for us to be able to hand over our project Facial Attendee.

This project is the result of the knowledge which has been acquired from the respective course. We tried our level best for preparing this report and project. The information of this report is mainly based on the implementation of Identifying, detecting and recognizing the Face. All of us gave our hundred per cent for making this report and project come together.

We, hope that you will find this plan worth reading. Please feel free for any query or clarification that you would like us to explain. Hope you will appreciate our hard work and excuse the minor errors. Thanking you for your cooperation.

Sincerely,
Mehdi Raza Rajani
k163904
Team Lead Facial Attendee

Abstract

Regularity in classes is known to be a vital factor in any students success. Most educational institutes go through the tedious process of attendance marking by manually calling out students IDs or circulation attendance sheets. These conventional methods apart from being time-consuming are also prone to errors (i.e. false attendance proxy attendance). Hence there is a need for fast and reliable attendance marking system. This report shows the software development process of an automated facial based attendance system namely Facial Attendee. The system is designed to be used by any organization to be used by their students and faculty separately to view and manage attendance respectively. This is completely based on deep learning neural network and implemented using Tensorflow framework. Real-time face recognition is done using FaceNet i.e. A Unified Embedding for Face Recognition and Clustering. The system Facial Attendee provides an effective solution to the hard process of conventional attendance marking. This system guarantees accuracy of 92 percentage and provides alternatives in case of any error.

Acknowledgement

First and foremost, we would like to offer this endeavour to the ALMIGHTY ALLAH for He bestowed upon us the wisdom, the strength, the peace of mind in order to complete this report. We would like to express our deepest appreciation to our course instructor Dr. Nauman Ghazi, who has the attitude of a genius, Dr. Tahir Qasim for assisting us in Facial Recognition related work, Ms Saeeda Kanwal for helping in AI related field and Ms. Samera Sultan for helping us in technical writing and proofreading our first draft. Due to there continuous and convincing efforts of conveying a spirit of adventure with regard to the project, and excitement with regard to teaching, we were able to complete this report/project. Furthermore, we are thankful to all our friends who helped us in giving images to train the data, and to our friends who provided us with feedbacks which helped in scrutinizing, reviewing, correcting and or adding/removing features.

Executive Summary

This report explains the design and implementation of a Facial Recognition Attendance System named Facial Attendee. Through this project risk of proxy attendance is nullified, attendance is marked with great ease and in less time, reporting feature also adds on the utility. This project has a one-time installation cost which protects the educational institute from daily paper and stationary cost. The attendance management is also done in an efficient manner which reduces the role of manager.

Contents

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Problem Statement . . . 1
1.3 Project Description . . . 1
1.3.1 Web Application for Admin . . . 2
1.3.2 Web Application for Teacher . . . 2
1.3.3 Mobile Application for Student . . . 2
1.4 Project Objective . . . 2
1.5 Project Justification . . . 2
1.6 Project Scope . . . 2
1.7 Planning and Project management . . . 3
1.8 Structure of the report . . . 5

2 Literature Review **6**
2.1 Introduction . . . 6
2.2 Importance of Attendance . . . 6
2.3 Neural Networks . . . 7
2.4 Template Matching . . . 7
2.5 Reference Face Graph for Face Recognition . . . 7
2.6 A Discriminative Feature Learning Approach for Deep Face Recognition . . . 8
2.7 Human face recognition technology . . . 8

3 Specification **9**
3.1 Requirements . . . 9
3.1.1 Functional Requirements - Web Application . . . 9
3.1.2 Functional Requirements - Mobile Application . . . 10
3.1.3 Non Functional Requirements . . . 10
3.2 Tools Used . . . 11

4 Project Design **13**
4.1 Block Diagram . . . 13
4.2 General Overview . . . 14
4.3 Architecture Design . . . 14
4.3.1 Architecture Design of Web Application . . . 14
4.4 Database Design . . . 15

5 Implementation **17**
5.1 Description of Whole Process . . . 17
5.1.1 Data Collection . . . 17
5.1.2 Data Preprocessing . . . 17
5.1.3 Training the Dataset . . . 17
5.1.4 Face Detection . . . 18
5.1.5 Face Recognition . . . 18
5.2 Reusable Components . . . 19
5.2.1 Django Admin Panel . . . 19
5.2.2 Sketch . . . 19
5.2.3 Bootstrap . . . 19
5.2.4 OpenCV . . . 19
5.2.5 Tensor Flow . . . 19
5.2.6 Volley for JAVA Android . . . 20

6 Testing and Evaluation **21**
6.1 Testing . . . 21
6.1.1 Unit Testing . . . 21
6.1.2 Component Testing . . . 21
6.1.3 System Testing . . . 22
6.1.4 User Testing . . . 22
6.2 Evaluation . . . 22
6.2.1 Evaluation of Face Detection . . . 22
6.2.2 Evaluation of Face Recognition . . . 23

7 Conclusions **25**

Appendices **28**

A Code Snippets **29**
A.1 Data Collection . . . 29
A.2 Data Preprocessing . . . 30
A.3 Data Training . . . 32
A.3.1 Train Main . . . 32
A.3.2 Classifier . . . 32
A.4 Identify Face . . . 34

B Prototype Designs **37**
B.1 Logo and Typography . . . 37
B.1.1 Logo . . . 37
B.1.2 Typography and Colors . . . 37
B.2 Mobile Application . . . 38
B.2.1 Sign In Page . . . 38
B.2.2 Home Page . . . 38
B.2.3 User Profile . . . 39
B.3 Web Page . . . 40

B.3.1 SignIn Page 40
B.3.2 Dashboard 41
B.3.3 Functionality Pages 41

List of Figures

1.1 Dependency Chart for Task 4
1.2 Sequence in which Task will be performed 4

3.1 Admin Use Case 10
3.2 Teacher Use Case 10
3.3 Student Use Case 10

4.1 Block Diagram of Project 13
4.2 ERD Diagram of Project 15

5.1 Original Image 18
5.2 Preprocessed Image 18
5.3 Haar Cascade Filter for Face Detection 18

6.1 Evaluation of Face Detection 23
6.2 Sample Face Detection 23
6.3 Evaluation of Face Recognition 24

B.1 Facial Attendee Logo 37
B.2 Typography and Colors 37
B.3 Sign In 38
B.4 Change Password 38
B.5 Home Page 38
B.6 Home Side Menu 39
B.7 Report Attendance 39
B.8 View Profile 39
B.9 Edit Profile 39
B.10 Registration Page 40
B.11 SignIn Page 40
B.12 Dashboard 41
B.13 Add Sheet Page 41
B.14 Reported Attendance Notifications Page 42
B.15 Pending Attendance Sheet 42

List of Tables

1.1 Task Division for Group Members 3
1.2 Structure of Report 5

6.1 Evaluation of Face Detection 22
6.2 Evaluation of Face Recognition 24

Chapter 1

Introduction

1.1 Motivation

Attendance is known to be a key factor in students academic performance. Almost 80 percentage of the educational institutes in Karachi follow the manual procedure of maintaining students attendance which is both time-consuming and prone to errors. Apart from this manual procedure, other procedures such as thumb verification or card validation also involve queuing delay. This report describes the working of the facial recognition attendance system namely Facial Attendee.

1.2 Problem Statement

This project aims to provide institutes to mark the attendance of their students a lot more conveniently and efficiently.
Taking manual attendance is a hard process, take time and cause a lot of paper-based work. This procedure is time-consuming and prone to errors; loss of sheets, updating error and dishonest attendance(proxy attendance). This leads to the idea of an accurate and fast attendance maintaining system.

1.3 Project Description

This project is a product for customers interested in automating their attendance system. This system is designed to provide Facial recognition system for the Attendance process. This system is largely cross-platform and is available to anyone using the departments provided computer resources in their organization. The system will run on a central server with each user having a remote user interface through a web browser to interact with it.
The project provides three deliverable:

- A web application for Admin
- A web application for teacher
- An Android application for students

1.3.1 Web Application for Admin

This application provides a web platform for course instructors to add, edit and delete teachers, students and courses. One can also allocate teachers to courses and enroll student to courses and teachers. This application is developed using Python Django framework.

1.3.2 Web Application for Teacher

This application provides a web platform for course instructors to maintain and view their course attendance. This application is developed using HTML-CSS.

1.3.3 Mobile Application for Student

This application developed using java android studio for students. Student can view their attendance information using this app. This application is developed using Java Android SDK.

1.4 Project Objective

The main purpose of the project is to bring in ease in the process attendance procedure and provide an effective facial recognition attendance solution.
The objectives include:

1. Reduce errors linked with traditional attendance marking procedure by providing an automated facial recognition system.
2. To provide students with a mobile application to keep track of their attendance.
3. To reduce time linked with manual attendance procedure, with the help of a camera to mark the attendance.

1.5 Project Justification

Several researchers have shown that absenteeism is directly related to bad academic performances [4]. This project serves the purpose of automating the tedious procedure of attendance marking and provides an effective solution to proxy attendance. This project will also reduce excess paper-loss and will contribute to healthy environment by saving Trees.

1.6 Project Scope

The project is targeted towards academic institutions providing a web portal for with all courses information along with respective attendance. The project also incorporates a mobile application for the students to view their attendance information course wise.

1.7 Planning and Project management

The Agile delivery framework was used in combination with the incremental approach. Due to the lack of time, we used an agile approach for Requirement gathering, UI UX Designing and Architecture design. Once it was finalized so we restricted further changes and focused on the defined architecture to develop the product.
We used Pair Programming technique to develop the product. The programming was divided into groups of two, for the major part of development. The first pair was responsible for web development. The UI/UX design was also done in pair by first making a pair Of UI/UX designer with a web developer for website and then an android developer for the Android application. Then Andriod Developer and Web Developer collectively design the Database. While at the same time the rest of the two members focused on face detection, template and Facenet formulation (training the data) and face detection.
The project was mainly divided into three phases System Architecture Design Phase, Face Detection Phase and Data training and Modeling Phase. The client will be involved during Requirement Gathering, UI UX Design and in Testing phase.
During the starting two task client can make changes and inform us but within a certain time period afterwards, the group will not entertain any request due to rapid delivery. Below is the task division for each group member which will be performed during this project.

Sr.	**Task Title**	**Task Description**	**Task Performer**
1	Requirement Gathering	Gather all the requirements	All
2	UI UX Design	Design mobile and web screens of system	Mehdi
3	Database Design	Design the database schema for the system with some test data	Tahir
4	Back-end API Writing	Design the REST API which will bring and post all the data	Tahir
5	Web Application Development	Design a web app for admin and teachers for performing the task	Ammar
6	Mobile Application Development	Design a mobile app for regular user for performing the task	Raza
7	Detecting a Face from a Photo	Detect a face from a image provided	Ammar
8	Template Formulation	Template will be formed for each user that will take all images input and sketch a figure that will latterly matched with image	Mehdi and Raza
9	Template Matching	Match a given image with all the template and predict the person	Tahir
10	Testing	Test the whole system regriously to detect some errors in system	All
11	Error Fixing	If some errors are detected then fix them and test them again	Respective Member

Table 1.1: *Task Division for Group Members*

Then we identified the dependency between the above task so we can plan accordingly. Below is the dependency graph for the above-mentioned task.

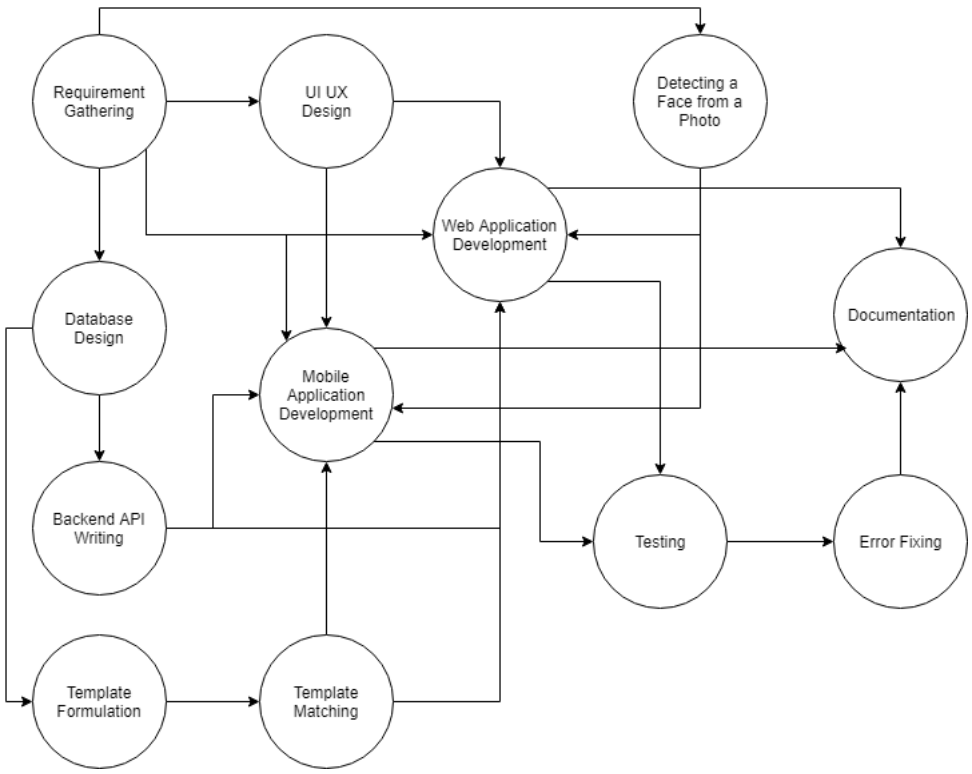

Figure 1.1: *Dependency Chart for Task*

On the basis of the dependency graph, we schedule our task accordingly which clearly signifies who we adequately allocated the task among the team.

Sr No.	Task Title									
1	Requirement Gathering	Independent								
2	UI UX Design		1							
3	Database Design		1							
4	Backend Api Writing			3						
5	Web Application Development			1	3	9				
6	Mobile Application Development			1	3	9				
7	Detecting a Face from a Photo			1						
8	Template Formulation			2						
9	Template Matching				7					
10	Testing						5	6		
11	Error Fixing								10	
12	Documentation							5	6	11

Figure 1.2: *Sequence in which Task will be performed*

1.8 Structure of the report

This project report consists of 7 sections. These sections are organized to show the development process of the project with architecture design, working followed by a conclusion.

Section Num.	**Description**
Section 1	Introduces the problem definition, project objectives, the contribution of this project, the scope of the work, and the project layout.
Section 2	Literature review of the related research paper
Section 3	Describes system specifications and how the requirements are met
Section 4	Discusses the design phase of the project including system architecture, database design, application design
Section 5	Defines the implementation phase of the project
Section 6	Evaluation and Testing are discussed in this section
Section 7	Concludes the whole project along with future recommendations

Table 1.2: *Structure of Report*

Chapter 2

Literature Review

2.1 Introduction

Face recognition is an important research problem applicable to a variety of fields and discipline. This aspect of computer vision is a bio-metric approach to identifying people based on their physical appearances. The process mainly includes detecting face from a scenario and matching it with the identical face present in the database with the help of an algorithm.
Facial recognition is now being applied in more and more industries; for banking in bio-metric verification, security monitoring, access control to restricted areas, attendance in organizations to even face unlock in smart phones.
The first method was put forth by Francis in 1988 [6]. The idea was to calculate norms of the facial curves and comparing it with the norms of existing information in the database. The next major breakthrough in facial recognition was by independently matching templates of three facial regions (eyes, nose and Mouth).
The three most common methods used today for facial recognition today are;

- Neural Networks
- Template Matching
- Reference Face Graph

2.2 Importance of Attendance

Bad academic performance is often linked with absenteeism. Researchers from around the world have been studying the impact of attendance on the overall performance of the students. The purpose of this literature review is to understand the relationship between class attendance and student performance and to understand whether the great emphasis on mandatory attendance is justified. It was generally assumed students cannot progress without being present during the lectures and learn from the classroom experience, until recently some researchers argued why some students failed to perform well despite being regular, hence contradicting the common perception regarding attendance and performance.
In autumn 2014, [4] studied the correlation between attendance and academic performance by keeping in consideration factors like bonuses, age group gender and bonus motivation.

The study showed a significant impact of absenteeism on performance with 84 percentage of the surveyed students strongly agreeing absenteeism negatively affect their GPA.

2.3 Neural Networks

The main idea behind the research paper, written by Dr Priya Gupta, Nidhi Saxena on Deep Neural Network for Human Face Recognition [7], lies behind the use of haar cascade for extracting facial features and feeding them instead of raw pixel values helps in decreasing the complexity of neural network based recognition framework as the number of redundant input features has been decreased. However an edition step of preparation of images to extract its features is added for each image but still, overall system performance is maintained especially for small datasets. Also, the use of DNN instead of Convolution Neural Network makes the process lighter and faster. Also, the accuracy is not compromised in the proposed method as average accuracy obtained is 97.05

2.4 Template Matching

Template matching is a simpler technique than CNN where an image is represented in a form a two-dimensional array of intensities and are compared on the basis of any metrics such as Euclidean distance. In the beginning, Brunelli[2] selected four templates eyes, nose, mouth and whole face. As time went on researchers set their templates accordingly. In this research paper [5] Anounand proposes wavelet transformation for preprocessing the face image and then extracting facial features.
Although template matching is simpler and logical to understand it has higher computational complexity.

2.5 Reference Face Graph for Face Recognition

This research paper [3] gives a solution for recognizing a face using the concepts of Reference Face Graph (RFG). RFG is a structure of nodes and the edges show the relationship between them. The nodes represent a single individual face with different variations. A basis descriptor of an image is a vector that describes the image in terms of its similarity to the reference faces. A reference face descriptor (RFD) is defined by incorporating the RFG node centrality metrics into the basis descriptor.
This method is composed of two steps. First, the reference face graph is built from various images of an individual. Secondly, Reference Face Descriptors are generated and it is compared with the test image. The RFD with the greatest similarity is returned as the recognized face. This method performs has a great accuracy up to 92.4 percentage on FREETE dataset and is unaffected by pose variation, misalignment and illumination variation. But expression variation affects the result of face recognition. The major drawback of this method is that it requires more training time but once training is done the system does not require the initial images because the input images are matched with the RFD.

2.6 A Discriminative Feature Learning Approach for Deep Face Recognition

This paper proposes [1] to enhance the discriminative power of the deeply learned features, by using convolutional neural network along with computer vision. The basic function which will introduce to enhance deeply learned features is centre loss which will respectively use for face recognition. The paper was aimed to develop a strategy function to improve the discriminative power of the deeply learned features. The researcher proposed a new loss function, referred to as centre loss by combining the centre loss with the softmax loss to jointly supervise the learning of CNN's, the discriminative power of the deeply learned features can be highly enhanced for robust face recognition. Experiments on several large-scale face benchmarks have convincingly demonstrated the effectiveness of the proposed approach, but this approach requires a large scale data beforehand in order to train the CNN.

2.7 Human face recognition technology

This research paper [8] summarizes 3 different existing technological methods of Human Face Recognition. The first method is an Artificial Neural Network for image classification. It is actually a human brain simulation. The method of the neural network for face recognition can train the image which has strong noise and part of defects, and the non-linear methods are sometimes more effective than the linear method. The earliest application of face recognition neural network takes advantage of the association of networks to memories people's face when the input image has a lot of noise or part of the image is lost, it can recall the exact face. Artificial Neuron cells are dedicated to performing the task of recognition. The second Method is SVM (Support Vector Machine). This method is a new pattern recognition method developed on the basis of statistical learning theory. It is based on the method of structural risk minimization principle. Statistical learning improves with time as more data increase statistical numbers hence it performs well on large data set. The last method, Algebraic characterization is taking a face image as a matrix, or by doing a linear projection matrix transformation can be extracted face features by use of the algebraic classification.it recognizance the human face image to the best to identify the characteristics; it is different from ANN Because of its inherent mechanism of paralleled computing and the overall storage of mode distribution.ANN has is comparatively more effective as it more accuracy even on the small data set.

Chapter 3

Specification

3.1 Requirements

3.1.1 Functional Requirements - Web Application

Login

This feature provides the instructor to log on to the website, this requirement is met using a validator function in the app to check the user against this registered teachers in the database.

Sign Up

This feature allows a new institute to get registered. This requirement is fulfilled by generating a request to the admin regarding the Institute registration.

Dashboard

After getting registered, instructor will land to Dashboard which will have summary regarding all of his / her courses. This Dashboard is implemented on the website using javascript library Chart.js to summarize the attendance using the Doughnut chart.

Take Attendance Page

This requirement is catered by using the python script of face recognition in the web application.

Pending Sheet

If the instructor selects 'Take Confirmation before Submitting Attendance' then this page will show all pending sheets.

Notifications

If students want to report an error in attendance, that reported attendance is shown in the notifications page.

3.1.2 Functional Requirements - Mobile Application

Login

This interface provides the student to log on to the application, this requirement is designed using XML and verifies student against the database before logging in.

Change Password

For the first time, user is asked to set the permanent password. Also, this screen can be used to change password screen when required to update the student password in the database.

Home

Shows the attendance record of all the courses enrolled of that day b requesting the attendance from the server.

View Profile

Shows details of students account.

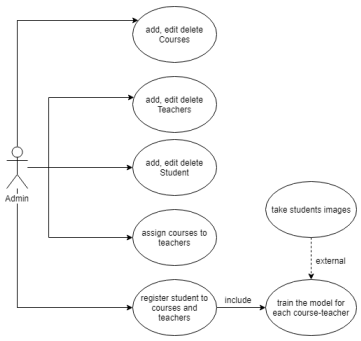

Figure 3.1: *Admin Use Case*

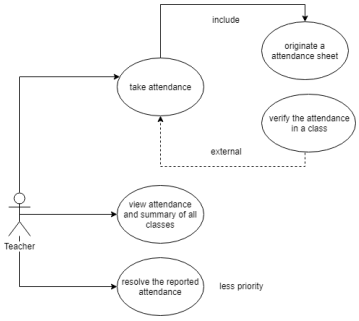

Figure 3.2: *Teacher Use Case*

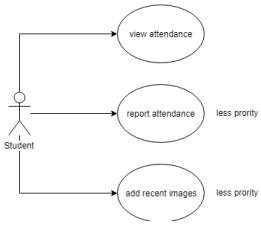

Figure 3.3: *Student Use Case*

3.1.3 Non Functional Requirements

Performance

The Facial Attendee systems server is built using the Django framework that provides a fast server-side solution.The system also uses FaceNet algorithm which is one of the most efficient face detection technique as mentioned in Literature Review. The system was specifically designed using python to ensure uniform compatibility across the system also resulting in overall better performance. The training done in this project was course wise to reduce the time in face recognition.

Safety

The system safety measures were taken beforehand to avoid any possible risk, the code was updated and versions maintained using GitHub, furthermore, systems development was regularly backed up on portable disks. Pair programming methodology was used to ensure the sharing of knowledge.

Security

User authentication functionality is made at the time of registration and logging in to avoid any unauthorized access to the system. The personal information of individuals is also secured as instructors only have access to students name id and courses. The system will not disclose personal information or pictures of the end users. The application will not communicate with any other devices or servers.

Accuracy

Accuracy was one of the motivations behind this project. Special emphasis was given to produce an accurate, trustworthy and reliable system. The target was to develop a system with at least 90 percentage accuracy, the system evaluation concluded that the developed system is 91.9 percentage accurate with as many as 10 students simultaneously being marked present.

3.2 Tools Used

Sketch

Sketch V54.1 was used to design the UI/UX of the web application and the Android application according to the Clients requirement.

ERDPlus

ERDPlus is a database modelling tool to quickly and easily create Entity Relationship Diagrams.This was used to design our systems database and show the foreign keys amongst different tables.

MySQL

MySQL is an open-source relational database management system. This relational database was chosen keeping in mind our requirement of structured data storage.

Django

Django framework was the most suited backend framework for Facial Attendee system because its python based, and the image detection and recognition scripts are written for this project are also in python, hence for easier compatibility Django was used. Django was also preferred over flask as it provides a default admin panel and its libraries are also helpful for APIs.

HTML/CSS/Bootstrap

Html, CSS and Bootstrap are also used in combination to develop the front end of the web application.

Java

Java was used to develop the native android app.

Github

Github was used to maintain version control of the system by systematically uploading all code on the repository.

Chapter 4

Project Design

4.1 Block Diagram

The below figure shows the general arrangement and basic flow of the designed system.

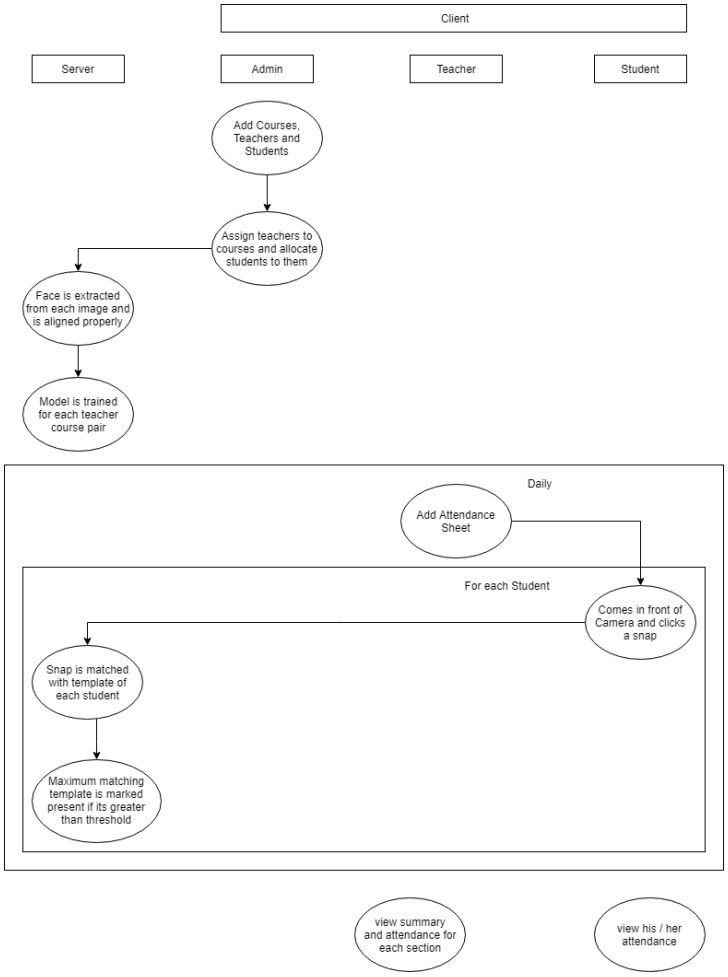

Figure 4.1: *Block Diagram of Project*

4.2 General Overview

The project has three different interfaces Admin, Teacher and Student. Initially, the admin will add teachers courses and students in the system. The model will be trained according to every section individually and select from the respective section to increase efficiency. Once this information is set into the system the teacher can add an attendance sheet for the particular day using the web application after logging in and selecting a course. Now the system will start recognizing the student and mark the attendance in the sheet. Once this process is completed the student can use android application to view his attendance details.

4.3 Architecture Design

The system architecture used in this project is Client-server architecture. Client-server architecture (client/server) is a network architecture in which each computer or process on the network is either a client or a server. Servers are powerful computers or processes dedicated to managing disk drives (file servers), printers (print servers), or network traffic (network servers). Clients are PCs or workstations on which users run applications. Clients rely on servers for resources, such as files, devices, and even processing power.

This architecture scheme is used to receive centralized services of trained model and database on the server side. The client's teacher interface(web application) and student interface (android application) can access this service by requesting it from the server. The client will be sending the pictures and the server will be recognizing the students.

4.3.1 Architecture Design of Web Application

The architecture scheme used within the web application is the Model View Controller(MVC). Its the default architecture provided by Django. Django follows a Model-View-Controller(MVC) architecture, which is split up into three different parts:

- The Model is the logical data structure behind the entire application and is represented by a database(generally relational databases such as MySql, Postgres).
- The View is the user interface - what you see in your browser when you visit a website. These are represented by HTML/CSS/Javascript files.
- The Controller is the middleman that connects the view and model together, meaning that it is the one passing data from the model to the view. With MVC, your application will revolve around the modeleither displaying it or manipulating it. MVC makes multiple views much easier by separating the views from the models.

4.4 Database Design

Following is the Entity Relation Diagram for our system.

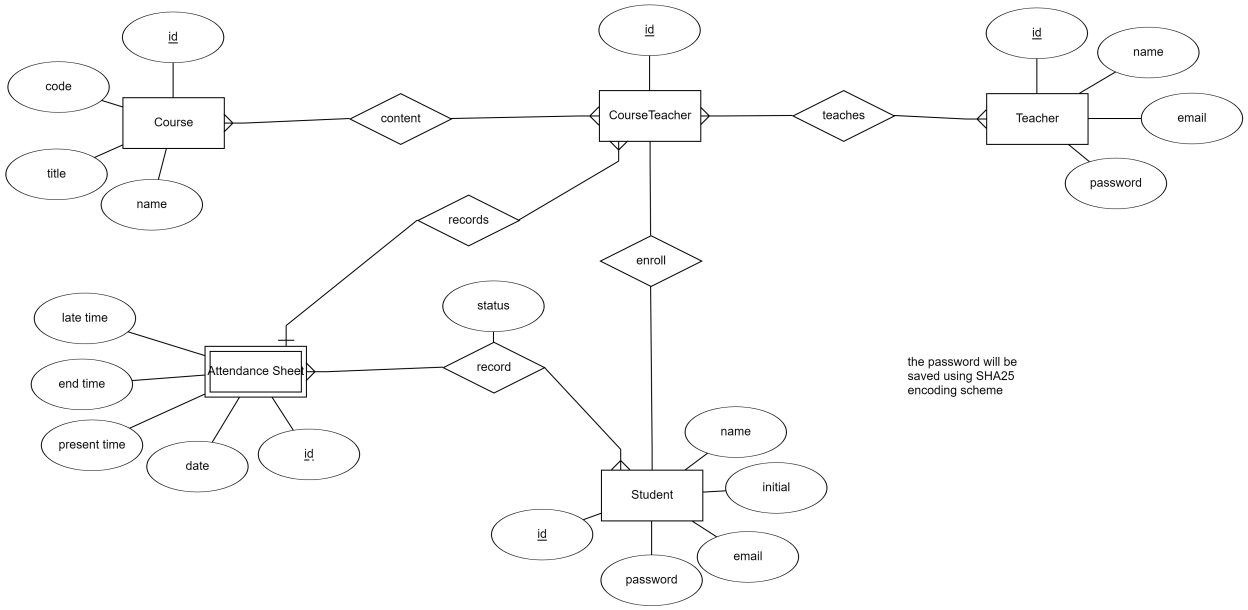

Figure 4.2: *ERD Diagram of Project*

The structure of the database is as following:

Student

This entity holds the information regarding registered students on the system. The primary key for this entity is its student's ID that is unique for every student. It has many to many relationships with the attendance sheet entity. Student data is used in attendance marking as well as in Android application for students.

Course

This entity has general information regarding course including course title, course code. This course entity is used for enrolled courses of student.Its primary key is the Course id.

Teacher

This table keeps information about teachers with their name and IDs. This information is used while signing in for teachers.

Course-Teacher

This entity is created when a course is assigned to any teacher.It takes foreign keys IDs from both Teacher and Course making a many to many relationships with each. The primary key for this table is CourseTeacher ID.

Attendance Sheet

This entity holds the record for every attendance taken. This stores information regarding the starting time of the class, end time and late time. CourseTeacher ID is taken as the foreign key in this entity.

Attendance

This table has the data for every students attendance (i.e. Absent, Present or Late) inside the attendance sheet.Attendance sheet ID is taken as the foreign key in this table and Attendance ID is the primary key.

Enrolled Course

This table specifies which students are enrolled in which courses.This entity takes both course ID and student ID as foreign keys.

Chapter 5

Implementation

5.1 Description of Whole Process

After the completion of Requirements Gathering and Analysis and Architecture Design, the Implementation phase started. The implementation stage was divided into the following phases:

5.1.1 Data Collection

This phase included the collection and arrangement of data in a proper format. During the collection phase of the implementation pictures of each student is taken and stored in files with students id (the unique id) as the title. This stage is critical as face recognition process will be based on this provided set of images. The key is to take images with a number of variations i.e. different angles, different emotions, with or without glasses etc.
We used below python script to get images of the person.

The code snippet is attached in Appendix A.

5.1.2 Data Preprocessing

After the collection process, the next step was to preprocess pictures to exclude unnecessary part of the image by cropping only the face out of the collected data. This cropping is done using a python script to quickly extract the faces from the collected data set.

The code snippet is attached in Appendix A.

Another process of preprocessing stage includes labeling each face image with folder name(i.e student id) this labeling information is stored in a text file.

5.1.3 Training the Dataset

This phase of implementation included training the model with the predefined model of OpenCV.

The code snippet is attached in Appendix A.

Figure 5.1: *Original Image*

Figure 5.2: *Preprocessed Image*

5.1.4 Face Detection

The implementation of face detection is done using Haar Cascade classifier using the Viola-Jones method of image detection.The Haar Cascade classifier is first trained to detect faces in the previous stage of implementation.This method uses small number of important features from a large set.The trained function of detection quickly looks for similar features and detects faces in the given picture.

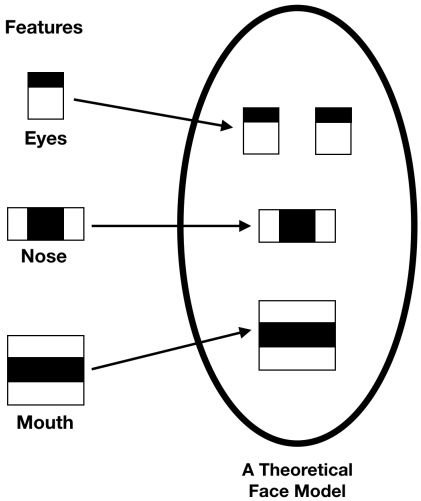

Figure 5.3: *Haar Cascade Filter for Face Detection*

The code snippet is attached in Appendix A.

5.1.5 Face Recognition

The face recognition in this project is implemented using Convolutions Neural Network, this strategy to recognize image is more accurate as compared to other methods. After the image is detected by the image detector Face recognizer uses those images to apply the algorithm and recognize the face based on the trained data model.

The code snippet is attached in Appendix A.

5.2 Reusable Components

5.2.1 Django Admin Panel

Django provides a default reusable admin panel to add delete modify the models of the database. In this system admin panel, the admin can add courses, add teachers and assign courses to teachers using the reusable Django admin panel. This admin panel saved the effort of creating views and templates explicitly for the admin. This reduced the development and designing time of frontend developer and UI/UX developer.

5.2.2 Sketch

Sketch provides some designs to be reused for the web and android application. In this system, Sketch designs were used and customized according to the clients requirements for the Web application and the Android application.

5.2.3 Bootstrap

Bootstrap provides reusable components such as navbar, buttons, footers and much more. In this project following bootstrap components were used;

- Navbar- Navbar of the templates of teacher interface were reused
- Buttons- Buttons of the bootstrap were also used in the website
- Tables- Table component is also used to display attendance of students with their IDs
- Footers- Sticky footer used to display a footer at the bottom of the page

5.2.4 OpenCV

OpenCV (Open source computer vision) is a library of programming functions mainly aimed at real-time computer vision. The library is cross-platform and free for use under the open-source BSD license. OpenCV supports the deep learning frameworks TensorFlow, Torch/PyTorch and Caffe. OpenCV-Python is a library of Python bindings designed to solve computer vision problems. By using these libraries we can reuse different functionalities and models which is already implemented in it.

5.2.5 Tensor Flow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices. TensorFlow computations are

expressed as stateful dataflow graphs. The name TensorFlow derives from the operations that such neural networks perform on multidimensional data arrays, which are referred to as tensors.

This reusable component enables us to train and test the images data using CPUs and GPUs effectively while using Neural Networks.

5.2.6 Volley for JAVA Android

Volley is an HTTP library that makes networking for Android apps easier and most importantly, faster. Volley offers the following benefits:

- Automatic scheduling of network requests.
- Multiple concurrent network connections.
- Transparent disk and memory response caching with standard HTTP cache coherence.
- Support for request prioritization.
- Cancellation request API. You can cancel a single request, or you can set blocks or scopes of requests to cancel.
- Strong ordering that makes it easy to correctly populate your UI with data fetched asynchronously from the network.
- Debugging and tracing tools.

Chapter 6

Testing and Evaluation

6.1 Testing

6.1.1 Unit Testing

Unit Tests were written to ensure the correct email address and phone number during the Sign Up process. Also, the correct ID selection for Foreign Key was ensured in Course-Teacher, Enrolled-Courses, Attendance and Attendance-Sheet Table.

6.1.2 Component Testing

Web Application

Manual testing was done for sign up and sign in to ensure security, 20 cases were tried with the accuracy of 100 per cent accuracy, these test cases verified;

1. Signup failure if the user account already exists
2. Sign in failure in case of wrong username or password
3. Sign in success if correct username and password are provided
4. Sign up success in case of a valid username and email and account not already created

Background separation

An automated python script was written to ensure that the face detector successfully separates the face from the background. This test was done for 20 people with a data set of 1000 pictures (50 pictures each).

Image recognition

This test was done to verify whether spectacles wearing person would be recognized without glasses or if the student got clicked during talking would he be recognized. The result was not up to the mark. This result was improved as a result of specific training the model with various angles, facial expressions while talking and with/without glasses.

6.1.3 System Testing

Mark Attendance

It was ensured that a correct person is marked present we have selected the best matching Facenet to the input picture and marked is present if and only if it is greater than minimum matching threshold.

Unavailability of Internet

In case of unavailability of the internet, the system will not crash and will prompt the user to get a stable internet connection this will guarantee against any inappropriate entry which may lead to problems.

6.1.4 User Testing

Alpha Testing

Alpha Testing: We are currently doing exhaustive testing while keeping the client on board from the internal members to identify any further flaws in the system. To eradicate all the loopholes will continue the alpha testing until we get satisfied.

Beta Testing

Still, we get satisfied we will initially launch our product as Beta Version in the market to test in the real world environment with true users. After the successful launch of Beta App, we will launch our final application which will guarantee flawless behaviour so people can build trust in it.

6.2 Evaluation

6.2.1 Evaluation of Face Detection

The system uses the Viola-Jones algorithm for face detection. In normal lighting conditions and quality, faces are detected efficiently as shown below. The system is designed for detecting multiple faces in a single picture but it is recommended to mark one student at a time. A summary of detecting accuracy with picture quality and a number of faces and the impact on accuracy is defined in the below table.

Image Size	**No. of Faces**	**Accuracy**
620 X 620	1	99.1%
620 X 620	4	98%
620 X 620	8	95%
620 X 620	10	91.9%

Table 6.1: *Evaluation of Face Detection*

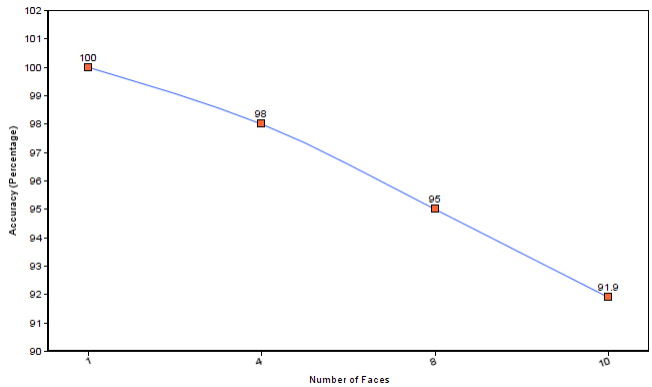

Figure 6.1: *Evaluation of Face Detection*

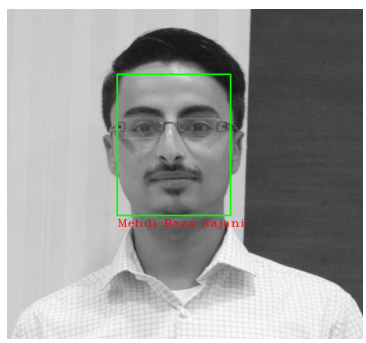

Figure 6.2: *Sample Face Detection*

The evaluation in the figure was done 20 times per case and the results showcased that system was less accurate with more number of faces, possible reason being merging of faces in small picture size. This evaluation, therefore, concludes that lesser people are preferred in face detection for better accuracy. Its hence evaluated that system successfully detects faces from the image and put boxes around detected faces as shown in the figure below.

6.2.2 Evaluation of Face Recognition

The system uses the Arxiv Algorithm to recognize a face from a group of data set that is cropped by the face detecting algorithm. In this section relationship between accuracy and number of data set for a person in the model as shown.
Below graph depicts that accuracy increases with respect to a number of pictures in the model. The graph indicates a steep change with data set of 45 and above hence recommending at least 50 pictures per student should be trained in the model.

Image Size	Dataset	Accuracy
620 X 620	5 - 15	64.4%
620 X 620	16 -25	65%
620 X 620	26 - 35	67%
620 X 620	36 - 45	70%
620 X 620	46 - 55	97%
620 X 620	56 - 65	97.1%
620 X 620	66 - 75	98%

Table 6.2: *Evaluation of Face Recognition*

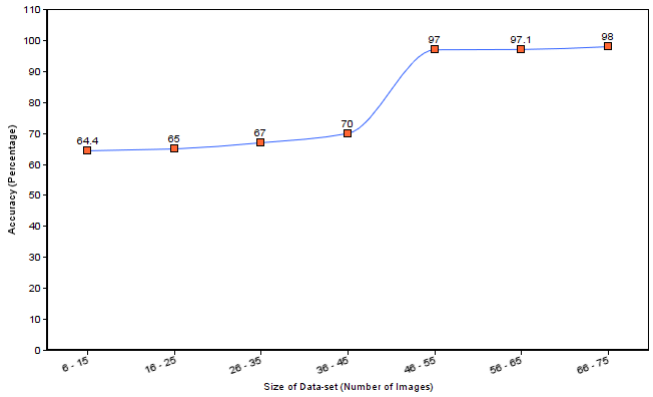

Figure 6.3: *Evaluation of Face Recognition*

Chapter 7

Conclusions

It is concluded that a reliable and efficient automated attendance management system has been been developed that can effectively replace a manual attendance system reducing time and paper cost. This system can be installed economically with a minimum of one camera and a computer in any educational organization. The system gives an accuracy of 92 percentage with a 4MP camera.

However a major drawback or limitations of this model. It takes 40-50 per person images with good quality of frontal face.

For future enhancement, a cross-check validation like thumb verification can also be integrated to ensure maximum accuracy. We can also self-train the model if it error is detected so that system accuracy can be increased up to a great extent.

Bibliography

[1] and. Research about human face recognition technology. In *2009 International Conference on Test and Measurement*, volume 1, pages 420–422, Dec 2009.

[2] R. Brunelli and T. Poggio. Face recognition: features versus templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10):1042–1052, Oct 1993.

[3] M. Kafai, L. An, and B. Bhanu. Reference face graph for face recognition. *IEEE Transactions on Information Forensics and Security*, 9(12):2132–2143, Dec 2014.

[4] Anna Lukkarinen, Paula Koivukangas, and Tomi Seppl. Relationship between class attendance and student performance. *Procedia - Social and Behavioral Sciences*, 228:341–347, 07 2016.

[5] Jenny Muir. Student attendance: Is it important, and what do students think? *CEBE Transactions*, 6:50–69, 09 2009.

[6] Karl Pearson. *PERSONAL IDENTIFICATION AND DESCRIPTION*, volume 3 of *Cambridge Library Collection - Darwin, Evolution and Genetics*, page 138216. Cambridge University Press, 2011.

[7] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Hybrid deep learning for face verification. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2013.

[8] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 499–515, Cham, 2016. Springer International Publishing.

Glossary

Terms	**Definition**
Attendance	The marking of number of students/workers for every class
Attendee	The student/worker enrolled/working in a course/organisation whose attendance is to be taken
Admin	The user of the web application; a person who is to take and maintain attendance
Brute Force Attack	Attacker submitting many passwords or pass-phrases with the hope of eventually guessing correctly
Encrypted	Data converted into the code to avoid unauthorized access
Enrols	To be officially registered in class (for attendance purpose)
MP	Mega Pixels
OpenCV	A library of programming functions mainly aimed at real-time computer vision
Overhead	Combination of excess or indirect computation time, memory, or any other resources
Recognize	To be able to identify or match the face from the database
SDK	Software Development Kit
Stakeholders	Any person who has interaction with the system who is not a developer
Tensorflow	A free and open-source software library for dataflow and differentiable programming
Web Portal	Web application for the teacher/manager of attendance

Appendices

Appendix A

Code Snippets

A.1 Data Collection

```
import cv2
import os,sys
import time

vidcap = cv2.VideoCapture(0)
success,image = vidcap.read()
count = 0
cwd = os.getcwd()
user_name = 'User Name'
dir = cwd + '/train_img/' + user_name
if not os.path.exists(dir):
    os.makedirs(dir)
success = True
while success and count!=50:
        print(count)
        time.sleep(0.75)
        success,image = vidcap.read()
        # save frame as JPEG file
        cv2.imwrite("train_img/"+user_name+"/frame%d.jpg" % count, image)
        count += 1
else:
    vidcap.release()
    cv2.destroyAllWindows()
    pass
```

A.2 Data Preprocessing

```
from scipy import misc
import os
import tensorflow as tf
import numpy as np
import facenet
import detect_face

class preprocesses:
    def __init__(self, input_datadir, output_datadir):
        self.input_datadir = input_datadir
        self.output_datadir = output_datadir

    def collect_data(self):
        output_dir = os.path.expanduser(self.output_datadir)
        if not os.path.exists(output_dir):
            os.makedirs(output_dir)

        dataset = facenet.get_dataset(self.input_datadir)
        with tf.Graph().as_default():
            gpu_options = tf.GPUOptions(per_process_gpu_memory_fraction=0.5)
            sess = tf.Session(config=tf.ConfigProto(gpu_options=gpu_options, log
            with sess.as_default():
                pnet, rnet, onet = detect_face.create_mtcnn(sess, './npy')

        minsize = 20  # minimum size of face
        threshold = [0.6, 0.7, 0.7]  # three steps's threshold
        factor = 0.709  # scale factor
        margin = 44
        image_size = 182

        # Add a random key to the filename to allow alignment using multiple pro
        random_key = np.random.randint(0, high=99999)
        bounding_boxes_filename = os.path.join(output_dir, 'bounding_boxes_%05d.

        with open(bounding_boxes_filename, "w") as text_file:
            nrof_images_total = 0
            nrof_successfully_aligned = 0
            for cls in dataset:
                output_class_dir = os.path.join(output_dir, cls.name)
                if not os.path.exists(output_class_dir):
                    os.makedirs(output_class_dir)
                for image_path in cls.image_paths:
                    nrof_images_total += 1
```

```
                    filename = os.path.splitext(os.path.split(image_path)[1])[0]
                    output_filename = os.path.join(output_class_dir, filename +
                    print("Image: %s" % image_path)
                    if not os.path.exists(output_filename):
                        try:
                            img = misc.imread(image_path)
                        except (IOError, ValueError, IndexError) as e:
                            errorMessage = '{}: {}'.format(image_path, e)
                            print(errorMessage)
                        else:
                            if img.ndim < 2:
                                print('Unable to align "%s"' % image_path)
                                text_file.write('%s\n' % (output_filename))
                                continue
                            if img.ndim == 2:
                                img = facenet.to_rgb(img)
                                print('to_rgb data dimension: ', img.ndim)
                            img = img[:, :, 0:3]

                            bounding_boxes, _ = detect_face.detect_face(img, min
                                                                        factor)
                            nrof_faces = bounding_boxes.shape[0]
                            print('No of Detected Face: %d' % nrof_faces)
                            if nrof_faces > 0:
                                det = bounding_boxes[:, 0:4]
                                img_size = np.asarray(img.shape)[0:2]
                                if nrof_faces > 1:
                                    bounding_box_size = (det[:, 2] - det[:, 0]) *
                                    img_center = img_size / 2
                                    offsets = np.vstack([(det[:, 0] + det[:, 2])
                                                         (det[:, 1] + det[:, 3])
                                    offset_dist_squared = np.sum(np.power(offsets
                                    index = np.argmax(
                                        bounding_box_size - offset_dist_squared *
# some extra weight on the centering
                                    det = det[index, :]
                                det = np.squeeze(det)
                                bb_temp = np.zeros(4, dtype=np.int32)

                                bb_temp[0] = det[0]
                                bb_temp[1] = det[1]
                                bb_temp[2] = det[2]
                                bb_temp[3] = det[3]

                                cropped_temp = img[bb_temp[1]:bb_temp[3], bb_temp
```

```
                    scaled_temp = misc.imresize(cropped_temp, (image

                    nrof_successfully_aligned += 1
                    misc.imsave(output_filename, scaled_temp)
                    text_file.write('%s %d %d %d %d\n' % (
                    output_filename, bb_temp[0], bb_temp[1], bb_temp
                else:
                    print('Unable to align "%s"' % image_path)
                    text_file.write('%s\n' % (output_filename))

        return (nrof_images_total,nrof_successfully_aligned)
```

A.3 Data Training

A.3.1 Train Main

```
import sys
from classifier import training

datadir = './pre_img'
modeldir = './model/20170511-185253.pb'
classifier_filename = './class/classifier.pkl'
print ("Training Start")
obj=training(datadir,modeldir,classifier_filename)
get_file=obj.main_train()
sys.exit("All Done")
```

A.3.2 Classifier

```
import tensorflow as tf
import numpy as np
import facenet
import os
import math
import pickle
from sklearn.svm import SVC
import sys

class training:
    def __init__(self, datadir, modeldir,classifier_filename):
        self.datadir = datadir
        self.modeldir = modeldir
        self.classifier_filename = classifier_filename

    def main_train(self):
        with tf.Graph().as_default():
            with tf.Session() as sess:
```

```
img_data = facenet.get_dataset(self.datadir)
path, label = facenet.get_image_paths_and_labels(img_data)
print('Classes: %d' % len(img_data))
print('Images: %d' % len(path))

facenet.load_model(self.modeldir)
images_placeholder = tf.get_default_graph().get_tensor_by_name("i
embeddings = tf.get_default_graph().get_tensor_by_name("embeddin
phase_train_placeholder = tf.get_default_graph().get_tensor_by_na
embedding_size = embeddings.get_shape()[1]

print('Extracting features of images for model')
batch_size = 1000
image_size = 160
nrof_images = len(path)
nrof_batches_per_epoch = int(math.ceil(1.0 * nrof_images / batch
emb_array = np.zeros((nrof_images, embedding_size))
for i in range(nrof_batches_per_epoch):
    start_index = i * batch_size
    end_index = min((i + 1) * batch_size, nrof_images)
    paths_batch = path[start_index:end_index]
    images = facenet.load_data(paths_batch, False, False, image_
    feed_dict = {images_placeholder: images, phase_train_placeho
    emb_array[start_index:end_index, :] = sess.run(embeddings, fe

classifier_file_name = os.path.expanduser(self.classifier_filenam

# Training Started
print('Training Started')
model = SVC(kernel='linear', probability=True)
model.fit(emb_array, label)

class_names = [cls.name.replace('_', ' ') for cls in img_data]

# Saving model
with open(classifier_file_name, 'wb') as outfile:
    pickle.dump((model, class_names), outfile)
return classifier_file_name
```

A.4 Identify Face

```
import tensorflow as tf
from scipy import misc
import cv2
import numpy as np
import facenet
import detect_face
import os
import time
import pickle
import sys

img_path='image.jpg'
aclassifier_filename = './class/classifier.pkl'
npy='./npy'
train_img="./train_img"

with tf.Graph().as_default():
    gpu_options = tf.GPUOptions(per_process_gpu_memory_fraction=0.6)
    sess = tf.Session(config=tf.ConfigProto(gpu_options=gpu_options, log_device_p
    with sess.as_default():
        pnet, rnet, onet = detect_face.create_mtcnn(sess, npy)

        minsize = 20  # minimum size of face
        threshold = [0.6, 0.7, 0.7]  # three steps's threshold
        factor = 0.709  # scale factor
        margin = 44
        frame_interval = 3
        batch_size = 1000
        image_size = 182
        input_image_size = 160

        HumanNames = os.listdir(train_img)
        # HumanNames = ['Ammar Rizwan', 'Mehdi Raza Rajani', 'Mujtaba Bawani', '
        HumanNames.sort()
        print(HumanNames)

        print('Loading feature extraction model')
        facenet.load_model(modeldir)

        images_placeholder = tf.get_default_graph().get_tensor_by_name("input:0"
        embeddings = tf.get_default_graph().get_tensor_by_name("embeddings:0")
        phase_train_placeholder = tf.get_default_graph().get_tensor_by_name("pha
        embedding_size = embeddings.get_shape()[1]
```

```
classifier_filename_exp = os.path.expanduser(classifier_filename)
with open(classifier_filename_exp, 'rb') as infile:
    (model, class_names) = pickle.load(infile)

# video_capture = cv2.VideoCapture("akshay_mov.mp4")
c = 0


print('Start Recognition!')
prevTime = 0
# ret, frame = video_capture.read()
frame = cv2.imread(img_path,0)

frame = cv2.resize(frame, (0,0), fx=0.5, fy=0.5)    #resize frame (optio

curTime = time.time()+1    # calc fps
timeF = frame_interval

if (c % timeF == 0):
    find_results = []

    if frame.ndim == 2:
        frame = facenet.to_rgb(frame)
    frame = frame[:, :, 0:3]
    bounding_boxes, _ = detect_face.detect_face(frame, minsize, pnet, rn
    nrof_faces = bounding_boxes.shape[0]
    print('Face Detected: %d' % nrof_faces)

    if nrof_faces > 0:
        det = bounding_boxes[:, 0:4]
        img_size = np.asarray(frame.shape)[0:2]

        cropped = []
        scaled = []
        scaled_reshape = []
        bb = np.zeros((nrof_faces,4), dtype=np.int32)

        for i in range(nrof_faces):
            emb_array = np.zeros((1, embedding_size))

            bb[i][0] = det[i][0]
            bb[i][1] = det[i][1]
            bb[i][2] = det[i][2]
```

```
                    bb[i][3] = det[i][3]

                    # inner exception
                    if bb[i][0] <= 0 or bb[i][1] <= 0 or bb[i][2] >= len(frame[0
                        print('face is too close')
                        continue

                    cropped.append(frame[bb[i][1]:bb[i][3], bb[i][0]:bb[i][2], :
                    cropped[i] = facenet.flip(cropped[i], False)
                    scaled.append(misc.imresize(cropped[i], (image_size, image_s
                    scaled[i] = cv2.resize(scaled[i], (input_image_size,input_im
                                           interpolation=cv2.INTER_CUBIC)
                    scaled[i] = facenet.prewhiten(scaled[i])
                    scaled_reshape.append(scaled[i].reshape(-1,input_image_size,
                    feed_dict = {images_placeholder: scaled_reshape[i], phase_tr
                    emb_array[0, :] = sess.run(embeddings, feed_dict=feed_dict)
                    predictions = model.predict_proba(emb_array)
                    print(predictions)
                    best_class_indices = np.argmax(predictions, axis=1)
                    # print(best_class_indices)
                    best_class_probabilities = predictions[np.arange(len(best_cla
                    print(best_class_probabilities)
                    cv2.rectangle(frame, (bb[i][0], bb[i][1]), (bb[i][2], bb[i][3
#boxing face

                    #plot result idx under box
                    text_x = bb[i][0]
                    text_y = bb[i][3] + 20
                    print('Result Indices: ', best_class_indices[0])
                    print(HumanNames)
                    for H_i in HumanNames:
                        # print(H_i)
                        if HumanNames[best_class_indices[0]] == H_i:
                            result_names = HumanNames[best_class_indices[0]]
                            cv2.putText(frame, result_names, (text_x, text_y), cv
                                        1, (0, 0, 255), thickness=1, lineType=2)
            else:
                print('Unable to align')
        cv2.imshow('Image', frame)

        if cv2.waitKey(1000000) & 0xFF == ord('q'):
            sys.exit("Thanks")
        cv2.destroyAllWindows()
```

Appendix B

Prototype Designs

B.1 Logo and Typography

B.1.1 Logo

Figure B.1: *Facial Attendee Logo*

B.1.2 Typography and Colors

Figure B.2: *Typography and Colors*

B.2 Mobile Application

B.2.1 Sign In Page

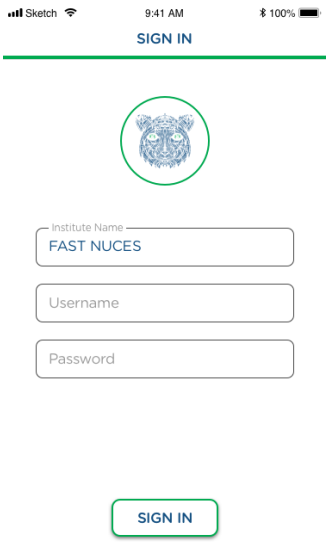

Figure B.3: *Sign In*

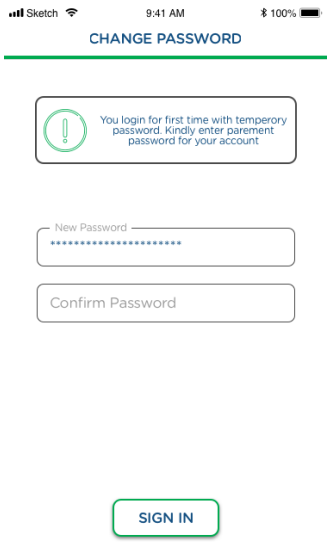

Figure B.4: *Change Password*

B.2.2 Home Page

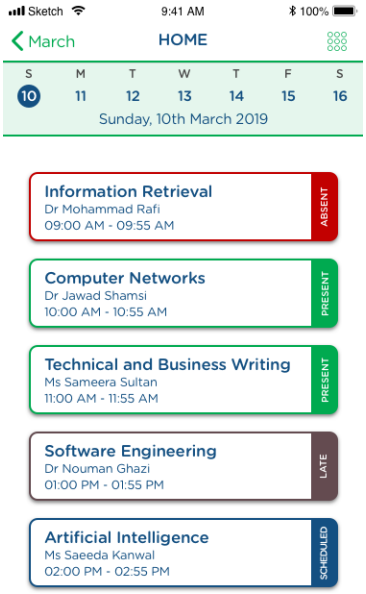

Figure B.5: *Home Page*

Figure B.6: *Home Side Menu*

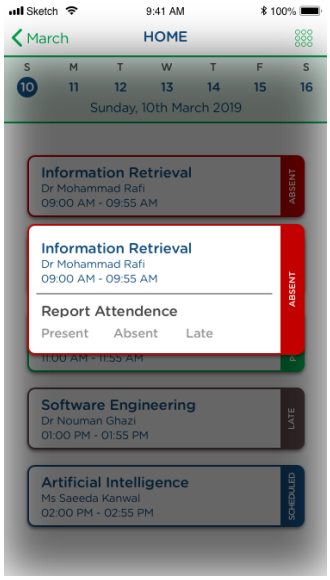

Figure B.7: *Report Attendance*

B.2.3 User Profile

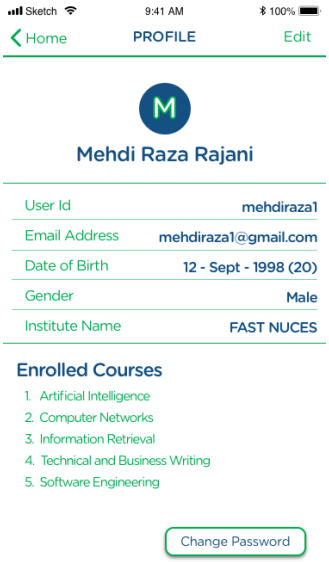

Figure B.8: *View Profile*

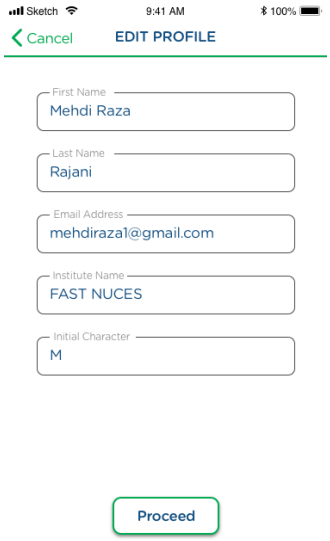

Figure B.9: *Edit Profile*

B.3 Web Page

B.3.1 SignIn Page

Figure B.10: *Registration Page*

Figure B.11: *SignIn Page*

B.3.2 Dashboard

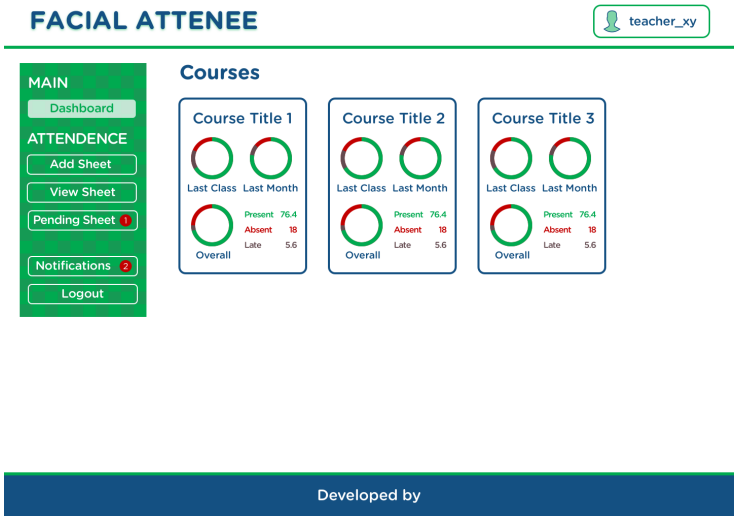

Figure B.12: *Dashboard*

B.3.3 Functionality Pages

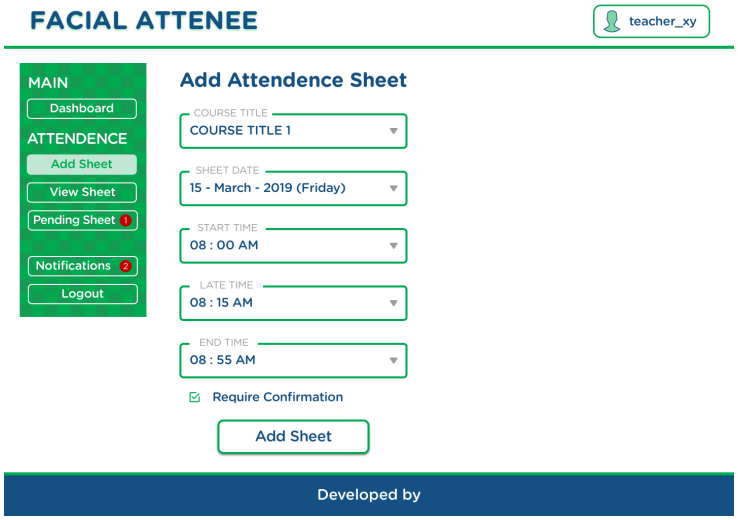

Figure B.13: *Add Sheet Page*

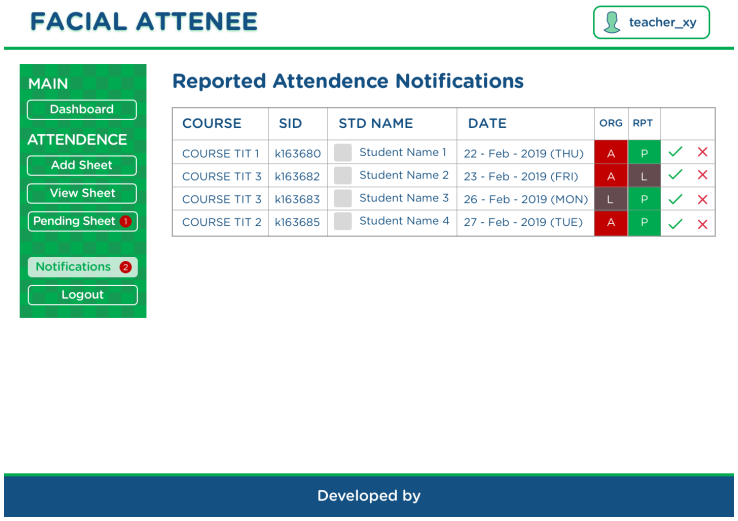

Figure B.14: *Reported Attendance Notifications Page*

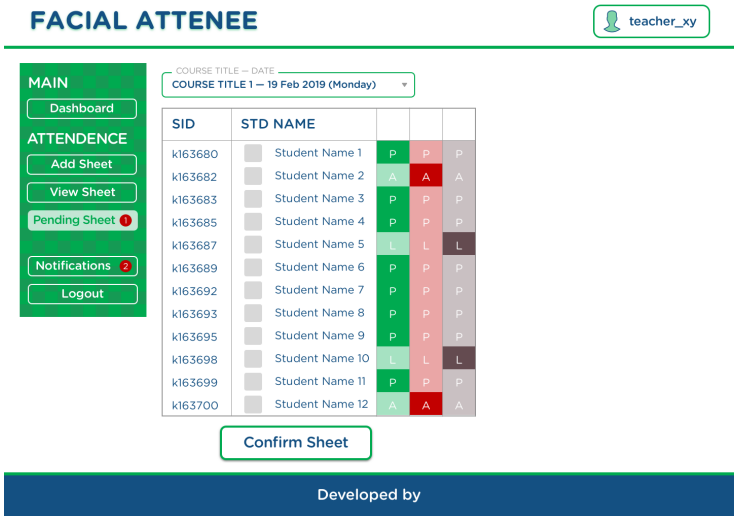

Figure B.15: *Pending Attendance Sheet*