A

Project Report

On

” Faculty Attendance System”

(CE353 – Software Group Project -IV)

Prepared by

18CE070 Dax Patel

18CE077 Jay Patel

18CE084 Ritul Patel

18CE086 Shail Patel

Under the Supervision of

Prof. Ronak Patel

Submitted to

Charotar University of Science & Technology (CHARUSAT)

for the Partial Fulfillment of the Requirements for the

Degree of Bachelor of Technology (B.Tech.)

in Computer Engineering (CE)

for 6th semester B.Tech.

Submitted at

CHARUSAT

CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

Accredited with Grade A by NAAC

Accredited with Grade A by KCG

U & P U. PATEL DEPARTMENT OF COMPUTER ENGINEERING

(NBA Accredited)

Chandubhai S. Patel Institute of Technology (CSPIT)

Faculty of Technology & Engineering (FTE), CHARUSAT

At: Changa, Dist: Anand, Pin: 388421.

May 2021.

DECLARATION BY THE CANDIDATES

We hereby declare that the project report entitled "**Faculty Attendance System**" submitted by us to Chandubhai S. Patel Institute of Technology, Changa in partial fulfilment of the requirement for the award of the degree of **B.Tech** in Computer Engineering, from U & P U. Patel Department of Computer Engineering, CSPIT/FTE, is a record of bonafide CE353 Software Group Project -IV (project work) carried out by us under the guidance of **Prof. Ronak Patel.** We further declare that the work carried out and documented in this project report has not been submitted anywhere else either in part or in full and it is the original work, for the award of any other degree or diploma in this institute or any other institute or university.

Dax Patel (18CE070)

Jay Patel (18CE077)

Ritul Patel (18CE084)

Shail Patel (18CE086)

Prof. Ronak Patel
Assistant Professor
U & P U. Patel Department of Computer Engineering,
CSPIT/FTE, CHARUSAT-Changa.

CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

Accredited with Grade A by NAAC
Accredited with Grade A by KCG

CERTIFICATE

This is to certify that the report entitled "**Faculty Attendance System**" is a bonafied work carried out by **Dax Patel (18CE070), Jay Patel (18CE077), Ritul Patel (18CE084), Shail Patel (18CE086),** under the guidance and supervision of **Prof. Ronak Patel** for the subject **Software Group Project -IV (CE353)** of 6th Semester of Bachelor of Technology in **Computer Engineering** at Chandubhai S. Patel Institute of Technology (CSPIT), Faculty of Technology & Engineering (FTE) – CHARUSAT, Gujarat.

To the best of my knowledge and belief, this work embodies the work of candidate themselves, has duly been completed, and fulfills the Partial requirement of the ordinance relating to the B.Tech. Degree of the University and is up to the standard in respect of content, presentation and language for being referred by the examiner(s).

Under the supervision of,

Prof Ronak Patel
Assistant Professor
U & P U. Patel Dept. of Computer Engineering.
CSPIT/FTE, CHARUSAT, Changa, Gujarat

Dr. Ritesh Patel
Head - U & P U. Patel Department of Computer Engineering,
CHARUSAT, Changa, Gujarat.

Chandubhai S. Patel Institute of Technology (CSPIT)

Faculty of Technology & Engineering (FTE), CHARUSAT

At: Changa, Ta. Petlad, Dist. Anand, Pin:388421. Gujarat.

Abstract

This project involves building an attendance system which utilizes facial recognition to mark the presence, time-in, and time-out of employees. It covers areas such as facial detection, alignment, and recognition, along with the development of a web application to cater to various use cases of the system such as registration of new employees, addition of photos to the training dataset, viewing attendance reports, etc. This report explains the libraries and machine learning based models and algorithms that have been used for facial detection and recognition. Explanation and use of Django, along with an SQLite database for web application development and database management has been provided. This project intends to serve as an efficient substitute for traditional manual attendance systems. It can be used in corporate offices, schools, and organizations where security is essential. The report also includes chapters covering project planning, methodology adapted and failures.

Acknowledgement

We are privileged to have this opportunity to express my gratitude and acknowledge everyone's never ending support and valuable contributions for our project. Prima facie, we would like to express my sincere gratitude to my mentor Prof. Ronak Patel for the continuous support of our project study and related research, for him patience, motivation, and immense knowledge. Our sincere thanks also go to Principal Sir Dr. A.D Patel, Dean Sir Dr. Amit Ganatra, HOD Sir Dr. Ritesh Patel who provided us an opportunity to work on a project and to be able to present the same. Last but not the least, we would like to thank my friends and family for supporting me spiritually throughout this project and for always being a constant source of inspiration. We also place on record, our sense of gratitude to one and all, who directly or indirectly, have lent their hand in this venture.

Table of Contents

List of Diagrams

List of Figures

Chapter 1: INTRODUCTION

1.1 Overview

Humans have always had an innate ability to recognize and distinguish between faces. Now computers are able to do the same, opening a plethora of opportunity for the advancement of human civilization. A face recognition system is a biometric technology used for mapping facial features and patterns for the purpose of identity storage and verification. It is being widely used to improve access and security, allows payments to be processed without physical cards. It also enables criminal identification, strengthening the pillars of law and enforcement. It also has started breaking natural barriers by helping blind or low vision users recognize faces and their surroundings.

1.2 Purpose

This project aims to automate the traditional attendance system where the attendance is marked manually. It also enables an organization to maintain its records like in-time, out time, break time and attendance digitally. Digitalization of the system would also help in better visualization of the data using graphs to display the no. of employees present today, total work hours of each employee and their break time. Its added features serve as an efficient upgrade and replacement over the traditional attendance system.

1.3 Methodology

The face recognition system analyses video feed frame by frame for detection of faces. Then a predictor is used to identify the facial landmarks in the faces detected using a predefined template. Using this template, the faces are aligned to increase the accuracy of the model. Now the unique embeddings of these aligned faces are generated. These embeddings are biometrics

of the face and hence unique. These embeddings are classified using the classifier. This classifier is trained on the images collected from the users beforehand to ensure the smooth workflow of the program.

1.4 Significance of the project

This project serves as a foundation for future projects based on facial detection and recognition. The report covers the libraries incorporated in this project for testing and optimization of machine learning algorithms. It also compares between algorithms with similar functionality. This project also covers web development and database management for a user-friendly UI.

Chapter 2: IMAGE PROCESSING WITH OPENCV

2.1 OpenCV Basics

OpenCV is an open-source library written in C++ that provides a set of programming functions that cater to real-time computer vision by making use of classic and state of the art Machine Learning and Computer Vision algorithms. Its applications include object detection, object tracking, 3D model extraction, face detection and recognition, scenery recognition, etc. OpenCV is used extensively in this project for webcam stream processing and image processing required for face detection, feature extraction, and alignment.

OpenCV can be installed by building from source which can be downloaded from OpenCV's website. Once it is downloaded, it can be imported using the command: import cv2 In order to load an image in OpenCV, the imread() method is used, in which the path of the image file is provided as a parameter. An image, once loaded, can be displayed using the imshow() method. Images are loaded as numpy arrays, and hence their dimensions can be retrieved using the shape attribute of numpy arrays. This provides another advantage - cropping images and extracting regions of interest is reduced to array slicing. Images can be resized using the resize() method. OpenCV loads images in BGR format which is not always desirable. Color space conversions to Grayscale and RGB formats is achieved easily using the cvtColor() method. All of these methods are used throughout the project for image manipulation.

2.2 Working with Video Streams

OpenCV provides VideoCapture and VideoWriter classes to capture videos and save them on disk, respectively. In this project, VideoCapture objects have been used to work with webcam streams. The VideoCapture() constructor takes one argument, which can either be a number, specifying the index of the camera device, or the path to a video file.

Once a VideoCapture object is created, it can be processed frame by frame, allowing us to apply the same image processing techniques that we use for static images. The video capture can be released with the release() method.

2.3 Imutils Library

Imutils is a python library that provides a series of convenience functions that make basic image processing tasks like translation, resizing, rotation and video stream processing a lot easier.

OpenCV and imutils is hence used in this project to capture videos via web camera, process them frame by frame, and transform those frames into a format that acts as a suitable input for face detectors, shape predictors, aligners, and models that extract embeddings.

Chapter 3: FACIAL DETECTION & ALIGNMENT

Face detection is the technology of recognizing human faces in a photo, video or live frame. Face detection can be said as a specific case of object-class detection. Face detection is the first step in the process of facial recognition. Face alignment, that can be thought of as data normalization, is a part of image preprocessing, wherein, after detecting the face, the image is cropped, rotated, scaled and adjusted so as to provide a normalized input to the face classifier.

3.1 Dlib's Detector

Dlib is a general-purpose cross-platform software library written in C++. Its libraries are available for use in different programming languages. The Object detection task can be achieved using Dlib's detectors such as Histogram of Oriented Gradients (HOG), or Convolutional Neural Network (CNN) based face detector.

3.2 Histogram of Oriented Gradients (HOG) Detector

HOG is a feature descriptor6 used in computer vision and image processing for the purpose of object detection. It works on the concept of finding gradients along the x and y-axis. These gradients are added vectorially and information such as magnitude and direction of the gradients are extracted forming edges in the image. Now the whole gradient image is divided into 8*8- dimension matrix i.e. carrying 64 gradient vectors. Each matrix contains 9 bins and each bin has a range of 40 degrees to allocate the gradient vector lying in that range.

After each gradient vector has been allocated a bin, a histogram of this matrix is formed which determines the overall direction and magnitude of the gradient. Luminosity plays a great role in the proper detection of gradients. To overcome this failure the value of the histogram is obtained in the red, green and blue frame and then normalized making it lighting invariant.

Finally, the normalized histogram is compared with its default histogram for faces to identify a face in an image.

3.3 Convolution Neural Network (CNN) based Detector

CNN is a category of Neural Networks that have proven very effective in areas such as image recognition and classification. A typical CNN, when provided with an input, applies one the following four main operations on it:

1. Convolution
2. Non-Linearity (ReLU)
3. Pooling or Sub Sampling
4. Classification (Fully Connected Layer)

The understanding of CNN requires time and is beyond the scope of this report. Dlib had a pre-trained CNN based facial detector available. The CNN based facial detector is able to detect faces at some very odd angles as well which wasn't the case with the HOG based detector. Though, as it takes a lot of time to work, it is not suitable for real-time face detection. It needs to be run on a GPU to get considerable speed. The results of comparison with the HOG based detector is shown in section below. The following code is used to initialize the CNN based face detector:

```
cnn_face_detector = dlib.cnn_face_detection_model_v1(args.weights)
```

3.3.1 HOG vs CNN

Dlib's CNN detector has far greater accuracy in detecting faces at odd angles as compared to its HOG detector. But due to it's complicated and computationally expensive algorithm it takes a lot longer than the HOG detector. Changing image resolution was found to

alter execution time drastically. These observations are evident in the following images. Since this project demands real-time face detection, Dlib's HOG detector was chosen over its CNN counterpart.

3.4. Facial Alignment using Landmark Predictors

Face Alignment is the process of identifying the geometric structure of faces in digital images and then attempting to obtain an alignment of the face based on rotation, scale, and translation. There are many methods to imply facial alignment. In this project, face alignment is achieved with the help of facial landmarks.

Facial detection is an important step before facial recognition as most face recognition algorithms can benefit highly from applying facial alignment and normalizing the input. Higher accuracy is obtained when face alignment is used.

Dlib has a 5 point and 68 points facial landmark predictor. It extracts important landmarks on the face like left eye start point, the center of the nose, etc. The facial landmark points help us to perform operations using the point coordinates. The operations performed for facial alignment are: -

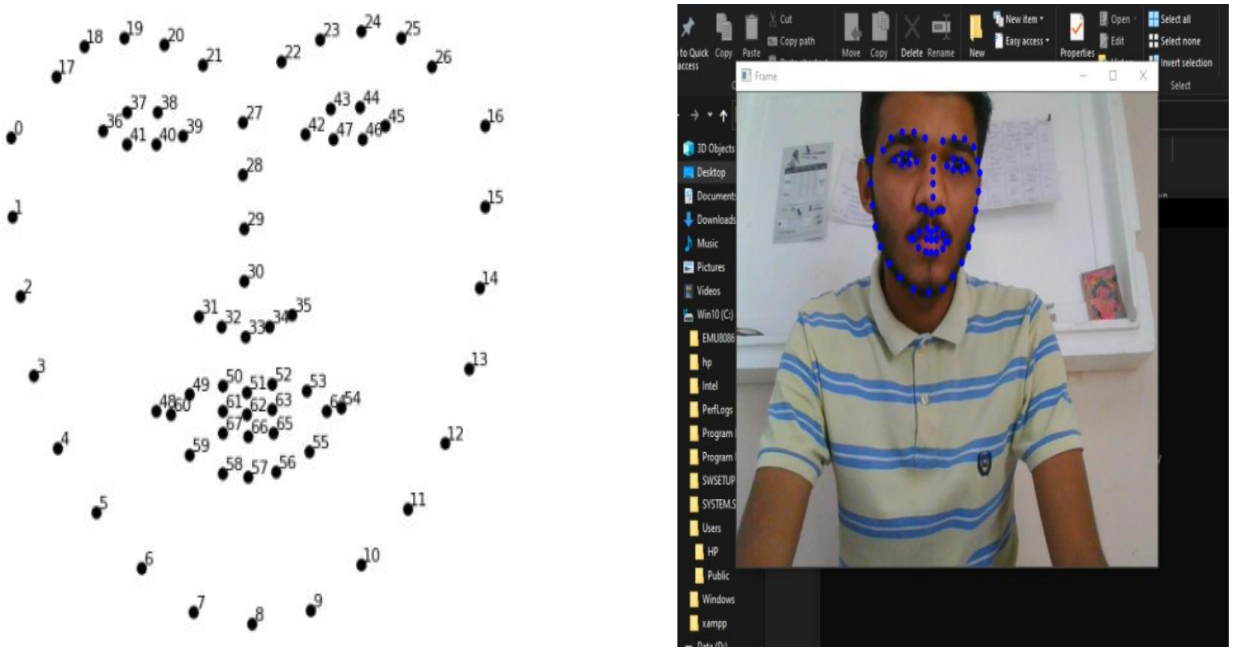
1) Rotation, such that the eyes lie on a horizontal line

2) Scaling, such that the size of faces is approximately identical

3) Translation, such that the image is centered

In this project, 68 points facial landmark predictor is used. Once a detector and a predictor are loaded into the system, the information in an image is forwarded to a face aligner which returns an aligned image. The aligned image is then fed into a CNN which extract face encodings, based on which the image is classified into a class.

```
predictor =
dlib.shape_predictor('./shape_predictor_68_face_landmarks.dat')
```



(a) (b)

Figure 1: Facial Alignment Results

Chapter 4: FACIAL RECOGNITION

A facial recognition system typically works by extracting face encodings from faces and comparing them with an existing database of known faces to find a match. Given an unknown image, the process of facial recognition comprises of detecting faces in that image, aligning them, extracting feature vectors from aligned faces and classifying each feature vector as belonging to one class from a set of classes. In order to extract feature vectors, or face encodings, the team used Convolutional Neural Network based models. Three different approaches were tested by the team:

1. Training & Testing of CNN model
2. OpenFace's NN4 Small2 inception model (pre-trained)
3. Face_recognition library by Adam Gaitgey

4.1 Training & Testing of CNN model

Once the CNN is built, it is trained on a number of classes, each having several training images. Once trained with images and associated labels, the CNN is expected to predict the associated label for an unknown image input. The training time of the CNN depends on the number of training images, the size of the batch, the number of times epochs, etc. In order to obtain high accuracy, it is necessary to train the CNN on a large dataset, which is computationally expensive and takes a lot of time. When trained and tested on a small dataset, the team was unable to get accurate results. Hence, for this project, the team decided to use pretrained CNNs to extract encodings from images.

4.2 NN4 Small2 Model

This model uses a deep convolutional neural network (CNN) of Google's Face Net architecture for feature extraction. Per as which uses TensorFlow as backend is used for

implementing the CNN. The CNN is used to extract 128-dimensional embeddings of faces from input images. In embedding space, Euclidean distance directly corresponds to a measure of face similarity. The last layer of CNN normalizes the embedding on the d-dim hypersphere and uses a triplet loss function to test the accuracy of the neural net classifying a face. Triplet loss algorithm aims to separate 2 different labeled images and bring same labeled image close in the embedding space. Thereby, creating clusters of same labeled images. This model showed an accuracy of 93.6% on the Labeled Faces in the Wild benchmark.

4.3 Face Recognition library

Face Recognition library works on deep metric learning10 tools to identify, train and classify test data. It is the world's simplest library for recognition and manipulation of faces. This model showed an accuracy of 99.38% on the Labeled Faces in the Wild benchmark indicated by python software foundation. The deep neural network used by it, represent the face on a 128-dimensional unit hypersphere. The embedding is a generic representation for anybody's face. Unlike other face representations, this embedding has the nice property that a larger distance between two faces embeddings means that the faces are likely not of the same person.

This property makes clustering, similarity detection, and classification tasks easier than other face recognition techniques where the Euclidean distance between features is not meaningful. This project has incorporated this library to extract the face encodings and face location from the images. Code for Generating embeddings using Face Recognition library:

```
X_face_locations = face_recognition.face_locations(face_aligned)
Faces_encodings = face_recognition.face_encodings(face_aligned, known_face_locations =
X_face_locations)
```

4.4. Classification

Classification belongs to the category of supervised learning where the targets/labels are provided with the input images. Given a set of labeled data points (here, known face encodings), a classifier is required to predict the class to which an unknown face encoding belongs.The process of prediction of the class to which certain data belongs using machine learning algorithms is called classification. There are two types of learners in classification as lazy learners and eager learners. Sklearn library in Python offers both lazy and eager classifiers like KNN and SVM respectively.

4.4.1 Lazy Learners

These classifiers simply store the training data and wait until testing data appear. When it does, classification is conducted based on the most related data in the stored training data. Compared to eager learners, lazy learners have less training time but more time in predicting. K nearest neighbors (KNN) is one of the lazy learners tested in this project. To predict a test image KNN looks into its k-closest neighbors and classifies the point as the majority class in those neighbors. It depends on two things: A metric used to compute the distance between two points and the value of 'k' the number of neighbors to consider. An optimized value of 'k' is required as a small value classifies the test data based on the closest neighbor and not the cluster. A large value of 'k' will be biased towards the class with a large dataset.

4.4.2 Eager Learners

These classifiers construct a classification model based on the given training data before receiving data for classification. It must be able to commit to a single hypothesis that covers the entire input space. Due to the model construction, they take a long time to train and less time to predict. Support Vector Machine (SVM) and CNN based classifiers are tested in this project.

Support Vector Machine is a linear model for classification. It can solve many linear and non-linear practical problems. The algorithm creates a line or a hyperplane which separates the data into classes. If the classes are inseparable in input space, the SVM plots these data points in a higher dimension using additional parameters. This higher dimensional space is called the feature 20 space. The feature space for face encodings is 128-dimensions. The test image is classified to the new cluster in the feature space.

4.5 NN4 Small vs Face Recognition Model

NN4 Small2 model generated encodings faster compared to the other model. The major drawback of this model was observed during clustering of generated encodings. The 128-D encodings were reduced to 2-D for graphical representation using t-Distributed Stochastic Neighbor Embedding (t-SNE). It was then observed that NN4 small2 model was unable to accurately distinguish between encodings of a different person, whereas dlib's face recognition was easily able to classify these encodings.

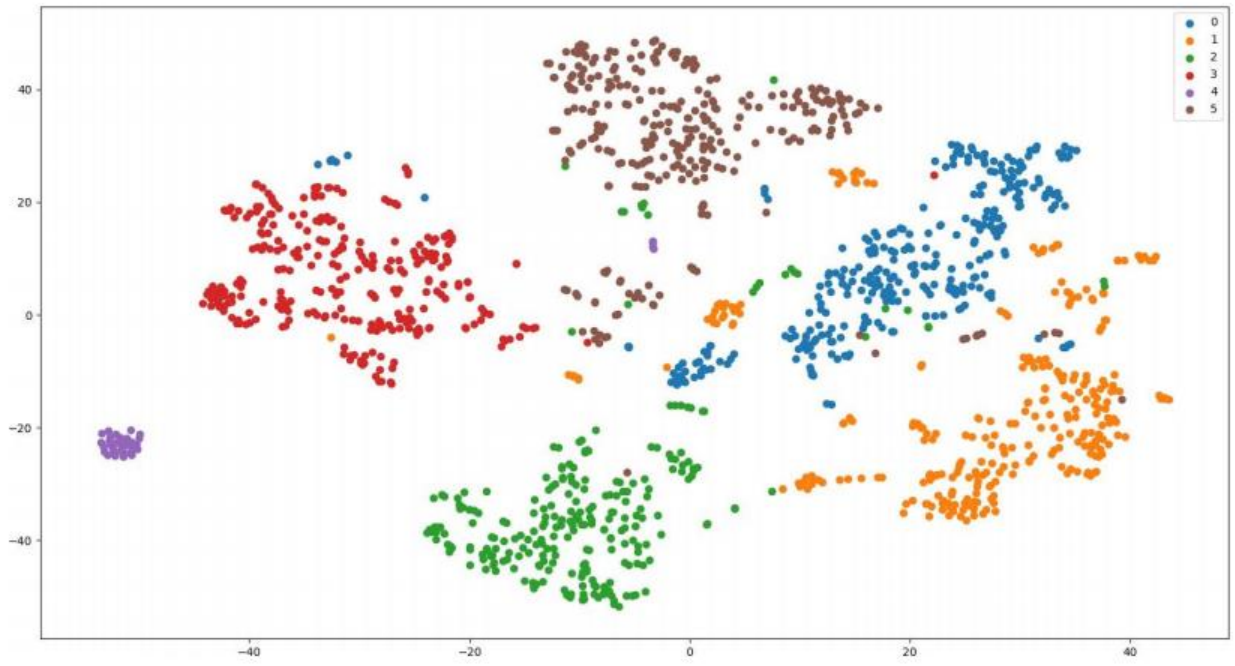


Figure 2: Clustering pattern representation facial encoding of 5 classes obtained using NN4 Small model

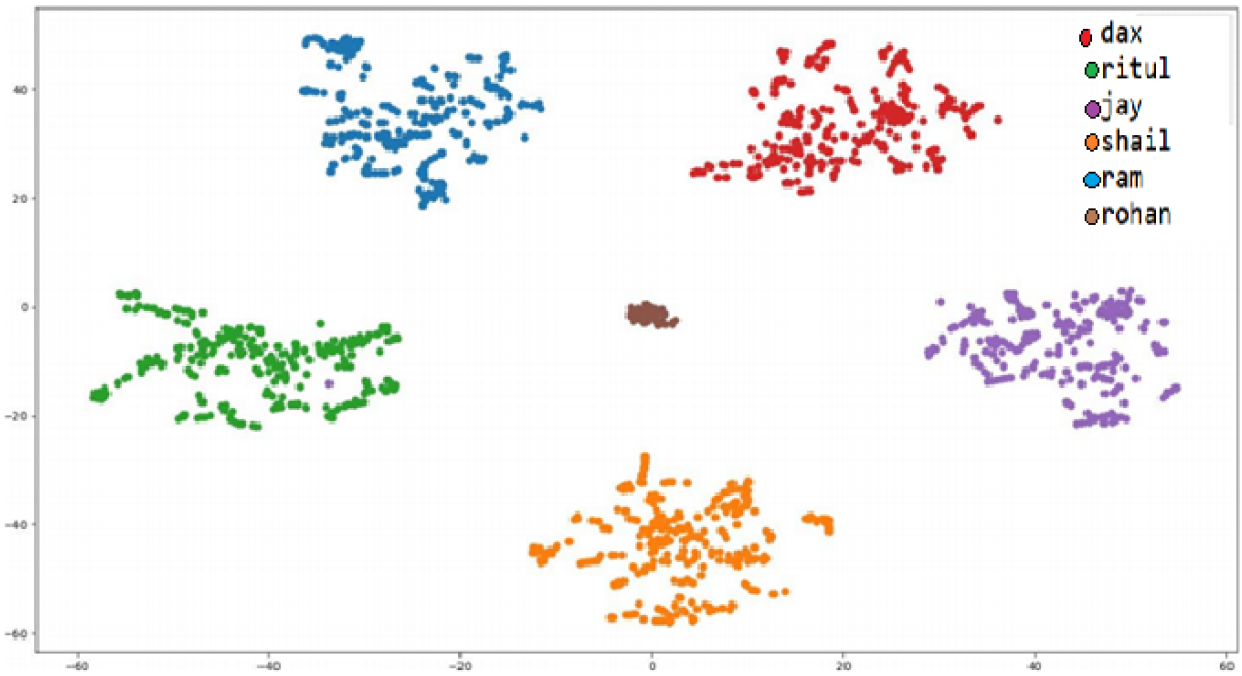


Figure 3: Clustering pattern representing facial encodings of 5 classes obtained using face_recognition Library

Chapter 5: WEB APPLICATION DEVELOPMENT & DATABASE MANAGEMENT

The project involved the creation of a web application that would work as an attendance system that uses the facial recognition model described in previous sections. For the development of this application, Django was used. Django is a high-level Python-based, an open-source web framework that allows the creation of websites with ease. It follows the Model-Template-View (MTV) Architecture (elaborated upon below). Along with being secure, portable, scalable, and versatile, it takes care of the intricacies and hassle of web development, so that the developer can focus solely on writing the application.

5.1 MTV Architecture

Every website has three main components- the input logic, the business logic, and the user interface. Traditionally, web development consisted of writing long lines of code in a single file, which served static web pages well. However, with larger, dynamic websites becoming the norm, this practice has become obsolete. Most modern websites are built on the MVC pattern which separates the three components into three different parts:

1. Model - The lowest layer which maintains data.

2. View-The top layer which presents the data to the user.

3. Controller- The layer that monitors the interaction between the model and the view.

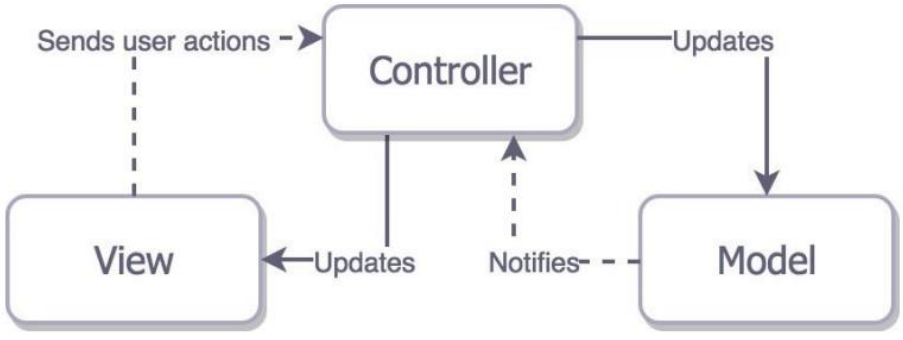


Diagram 1: Model View Controller Pattern

Django's MTV architecture is a slight variation of the MVC pattern. The model maintains data, the template acts as the presentation layer, and the view formats the data received from the model and sends it to the template to be displayed. It also processes the input by the user and sends create, update, delete requests to the model. The template layer in Django is akin to the view layer in the typical MVC and consists of HTML, CSS and Javascript code.

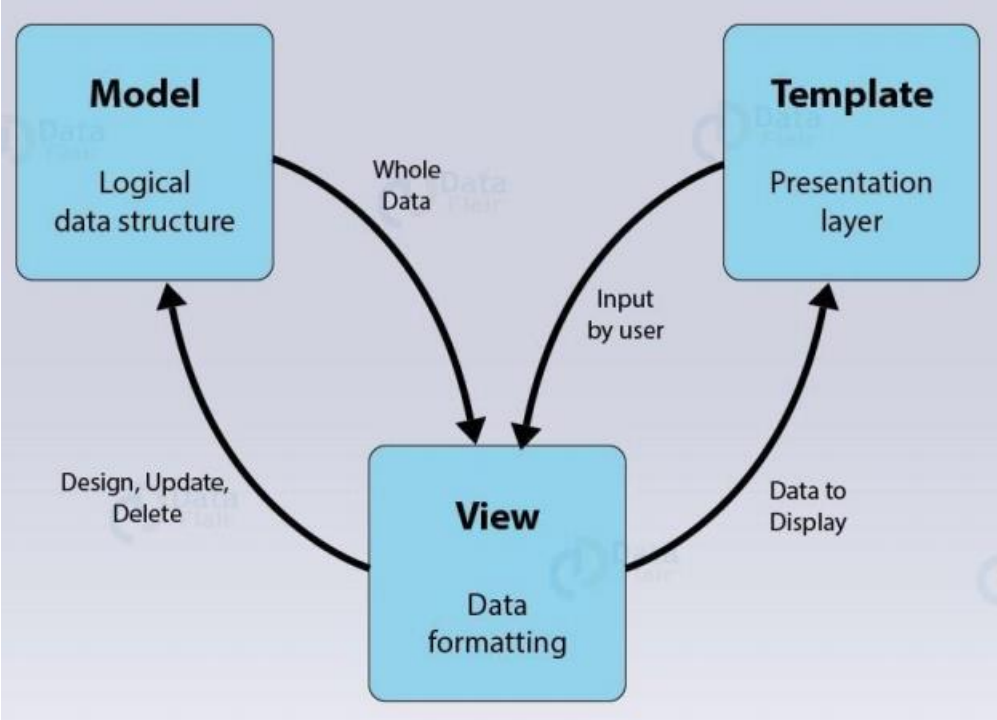


Diagram 2: Model Template View Attendance

Django makes it easy to communicate between the model, the view, and the templates eliminating the need for writing complex code to achieve the same.

5.1.1 Installation

Django is installable via pip- the python package installer. To install Django, the following command needs to be written on the terminal:

pip install Django

5.2 Project Structure

The command django-admin startproject my_project followed by django manage.py startapp my_app is used to initialise a django project named ‘my_project’ consisting of one application named ‘my_app’. The project thus has the following structure:

myproject/
manage.py
myproject/
    init .py
    settings.py
    urls.py
    Wsgi.py
    myapp/
        init .py
        admin.py
        models.py
        tests.py
        views.py

The views, models, and templates are thus separated, allowing us to harness the power of MTV architecture. The models.py file is where we define our database models. The templates contain HTML files which create the UI layer. The views.py file is used to communicate between the models and the templates. It consists of different functions, or views, which take in web requests and return web responses. As a result, the HTML contents of the associated template, along with linked CSS and JS files, are displayed as a web page, at the url mapped to that view. This url mapping is specified in the urls.py file. Django views are linked with their corresponding 25 contents using the render function, and data to be displayed is passed using the Django templating language.

5.3 Databases, ORM & Models

Django allows developers to connect their applications with databases such as SQLite, MySQL, PostgreSQL. Every database implements and uses SQL in a different manner. Django takes care of the hassle of generating SQL specific to that database by providing you with the ability to write Python data structures instead of the database language. Django includes an Object Relational Mapping (ORM) layer that can be used for this interaction with data from various relational databases like SQLite, PostgreSQL. Django also provides Models, which are Python classes which act as a bridge between the database and the server. The following figure illustrates how the Django ORM and Models, together implement traditional tables, rows and columns as classes, objects, and attributes, making data a lot easier to work with.

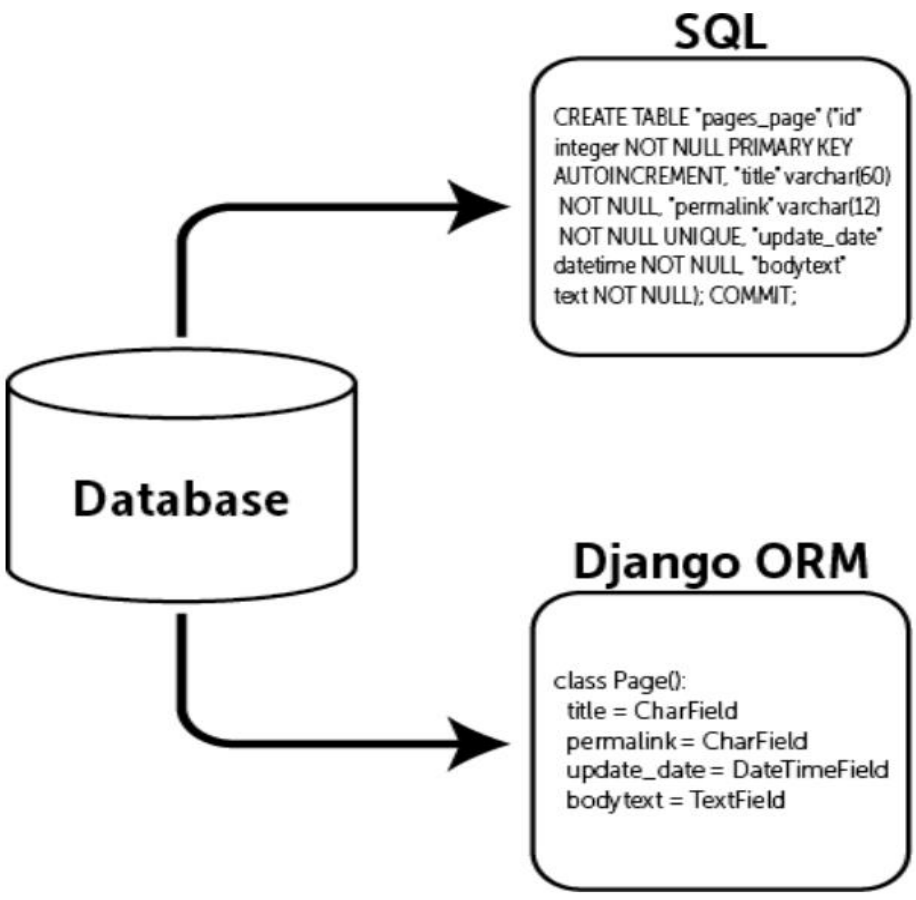


Diagram 3: Django's ORM

Data can now be fetched, created and saved as objects, and Django takes care of the SQL for the developers. This project usesthe built-in User model provided by Django, which storesthe information of all employees. This model can be imported using the following command: from django.contrib.auth.models import User

This project also uses four other model classes 'Present' and 'Time' that store Attendance (present/absent) and time-in, time-out, respectively of all employees and other two model classes are 'User_Details' and 'Holiday' that store the details of the employee and list of holidays. The following code is used to create these four models:

```
class Present(models.Model):
    user=models.ForeignKey(User,on_delete=models.CASCADE)
    date = models.DateField(default=datetime.date.today)
    present=models.BooleanField(default=False)

class Time(models.Model):
    user=models.ForeignKey(User,on_delete=models.CASCADE)
    date = models.DateField(default=datetime.date.today)
    time=models.DateTimeField(null=True,blank=True)
    out=models.BooleanField(default=False)
    image=models.BinaryField(blank=True)


class User_details(models.Model):
    user=models.ForeignKey(User,on_delete=models.CASCADE)
    #user=models.CharField(max_length=100)
    email=models.CharField(max_length=120)
    contact=models.CharField(max_length=10)
    department=models.CharField(max_length=100)

class Holiday(models.Model):
    name=models.CharField(max_length=30)
    date=models.DateField()
```

The user field acts as a foreign key to the User model. The date field stores the current date. The present field is a boolean field that stores True for present, and False for absent. The time field stores the time of entry or exit, depending upon the value of the outfield, which stores True for time-out, and False for time-in.

5.4 Front End with Bootstrap & CSS

Django is a very powerful framework, but it can't be used for front end development. In order to develop the website's UI and UX, Bootstrap and CSS styling were used. These CSS files were stored in a separate static folder within the Django project. CSS- Cascading Style sheets are used for describing the presentation of a document written in a markup language such as HTML. It allows the addition of styles to a webpage with the use of classes and IDs. Bootstrap is a front end framework that simplifies the process of adding styles, by providing built-in CSS classes that 28 can be added to HTML elements, allowing the developer to forgo the process of writing complicated CSS.

For this project, Bootstrap was added via the Bootstrap CDN. The url of Bootstrap CSS files is simply added within the tag of the HTML file, and Bootstrap classes can then be used throughout the file. The Bootstrap typography, navbar, table, button, card, container, alert classes were used extensively throughout the project.

5.5 Django Template Language

Django Template Language is used to send data from the view to the templates. It provides the developer with tags that function similarly to some programming constructs - such as the 'if' tag for boolean tests and the 'for' tag for looping. The templating language has been used throughout the project for interaction between the view and the templates.

5.6 Tabular Representation of Attendance Data

The application provides the admin with the feature to view attendance reports, where in attendance data is displayed as tables. This is achieved via Django's ORM and Templating Language. Data is fetched from the attendance database in the form of querysets, making use of

Django's ORM. These querysets are then passed to the template (HTML) files using Django's templating language. These query sets can be displayed as a table by using the {% for tag %} of the templating language for displaying rows and outputting each attribute of the object in a different table cell (column). 29 The table formed using queryset and its attributes are displayed in the following

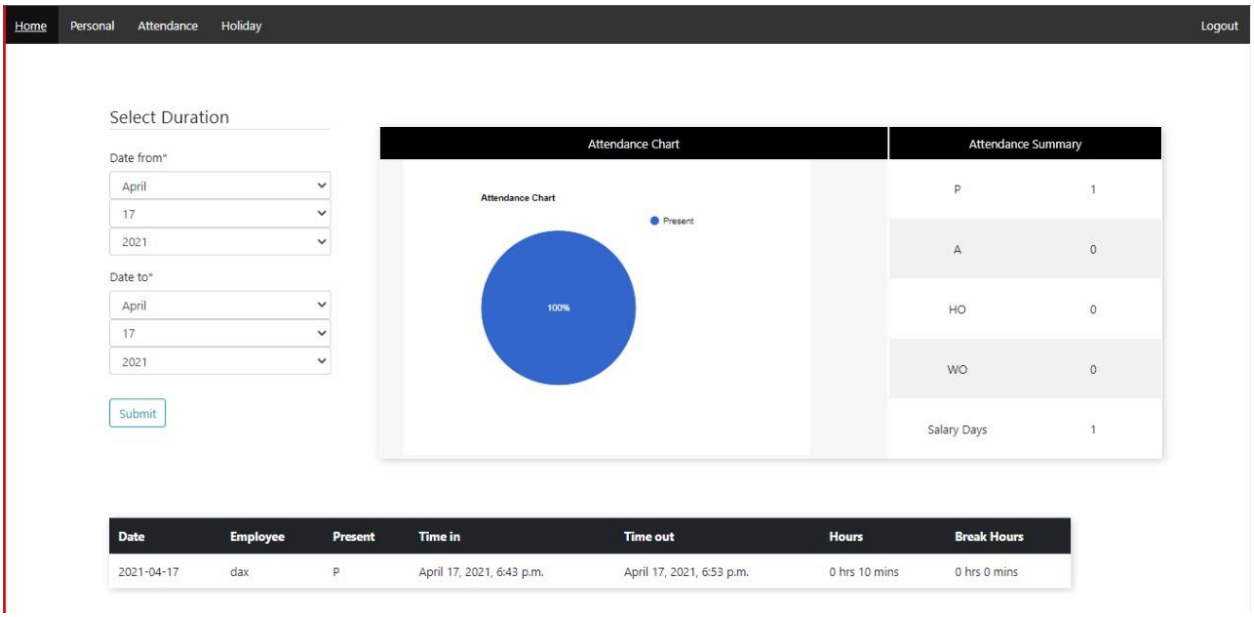


Figure 4: Tabular Representation of Attendance Data

Chapter 6: DATA VISUALIZATION

One of the features of this application is that it allows the admin to view attendance data in the form of graphs and plots. For this purpose, Matplotlib and Seaborn libraries have been used. Data is first fetched from the database in the form of query sets, using Django. Query sets can be converted to Pandas dataframe objects using the django_pandas library. These data frames are then used as direct inputs to Seaborn graphing functions.

6.1 Matplotlib & Seaborn

Matplotlib is a 2D Python 2D plotting library that can be used to generate plots, histograms, power spectra, bar charts, error charts, scatterplots, etc. from datasets. Seaborn is a graphical library built on top of Matplotlib. Seaborn provides a high-level interface to draw statistical graphics. Seaborn functions operate on data in the form of Pandas dataframes. Both Matplotlib and Seaborn are pip installable.

6.2 Examples

1. A Seaborn bar plot demonstrating the number of hours worked by each employee on a particular date is displayed in the following figure (see Appendix D for code on how django_pandas library and Seaborn can be used to plot data from the database).

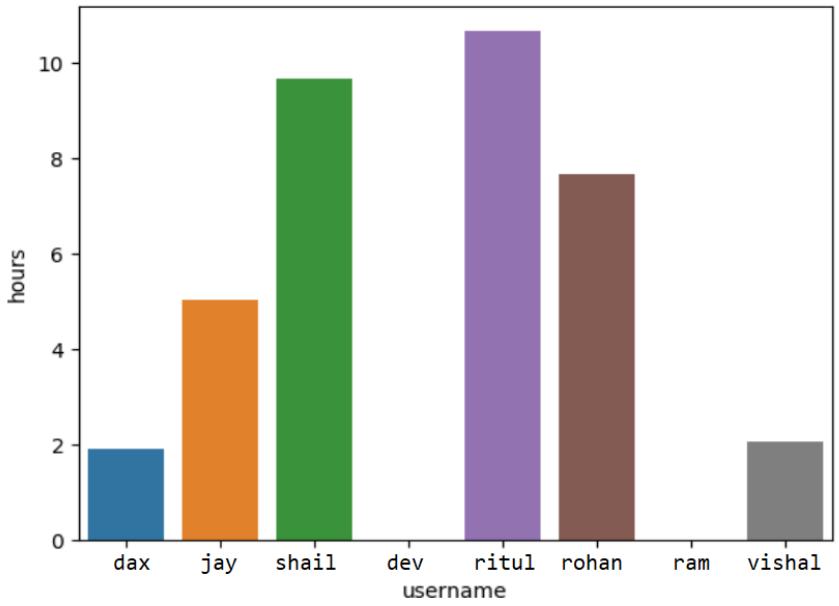


Figure 5: Hours worked vs username for a particular date

2. A Seaborn line plot demonstrating the number of hours worked by a particular employee for a range of dates is displayed in the following figure. Note that it allows easy visualization of the weekly minimum.

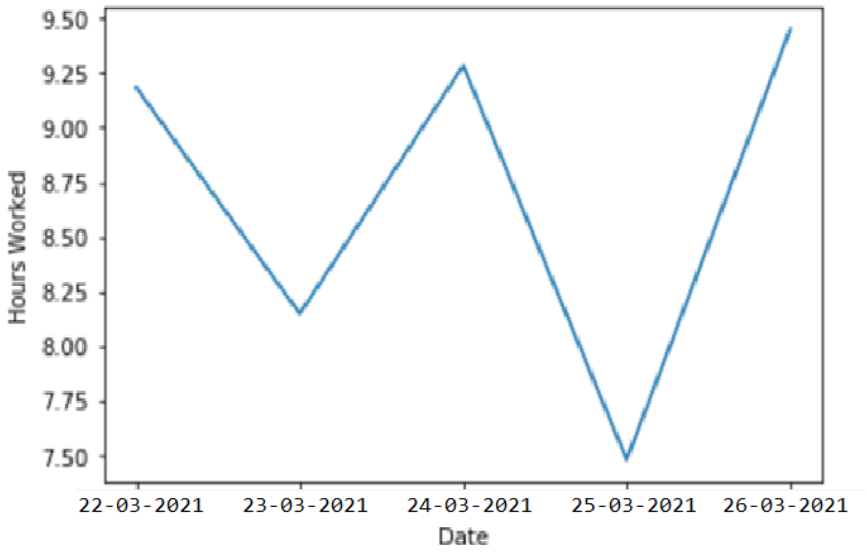


Figure 6: Hours worked vs Date for a particular employee

3. Seaborn line plots demonstrating the number of employees present each day of the week are displayed below.

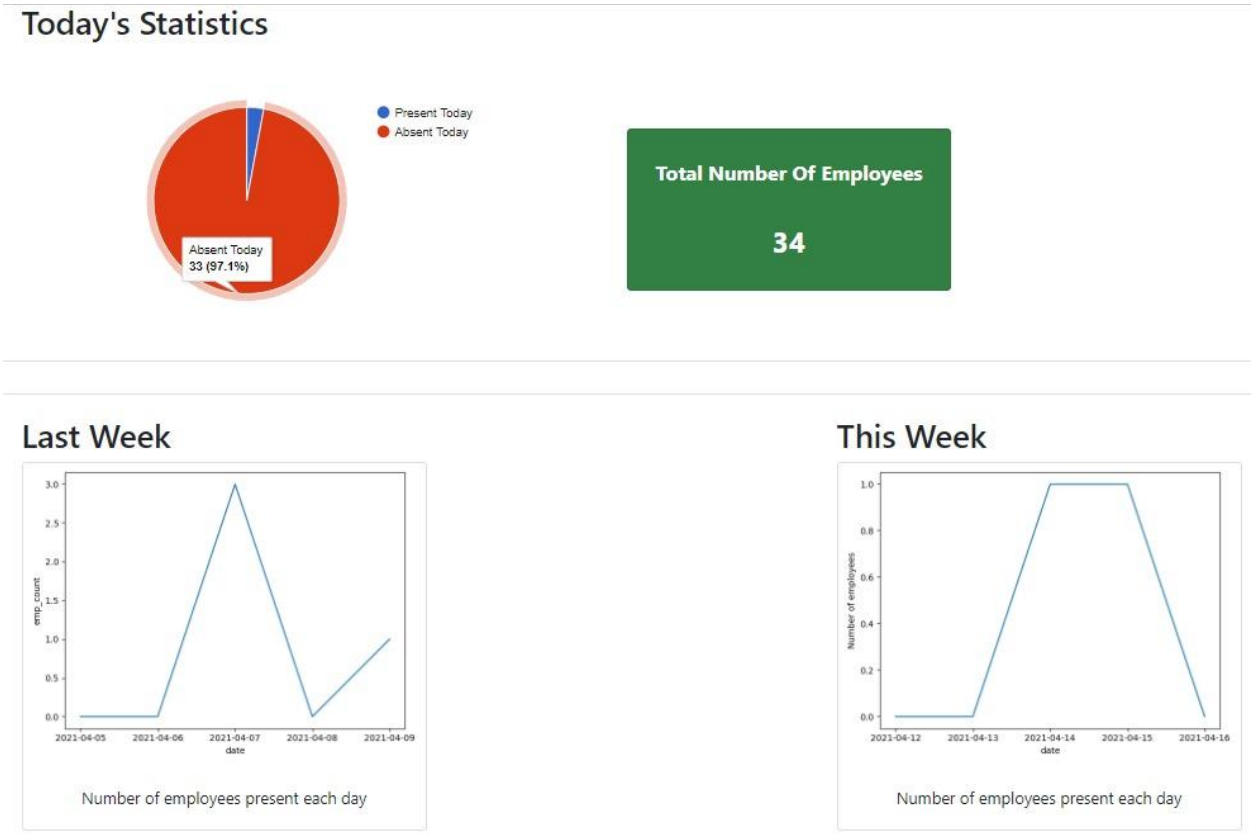


Figure 7: Total Number of Employees vs Date

Chapter 7: SYSTEM DESIGN

7.1 Activity Diagram

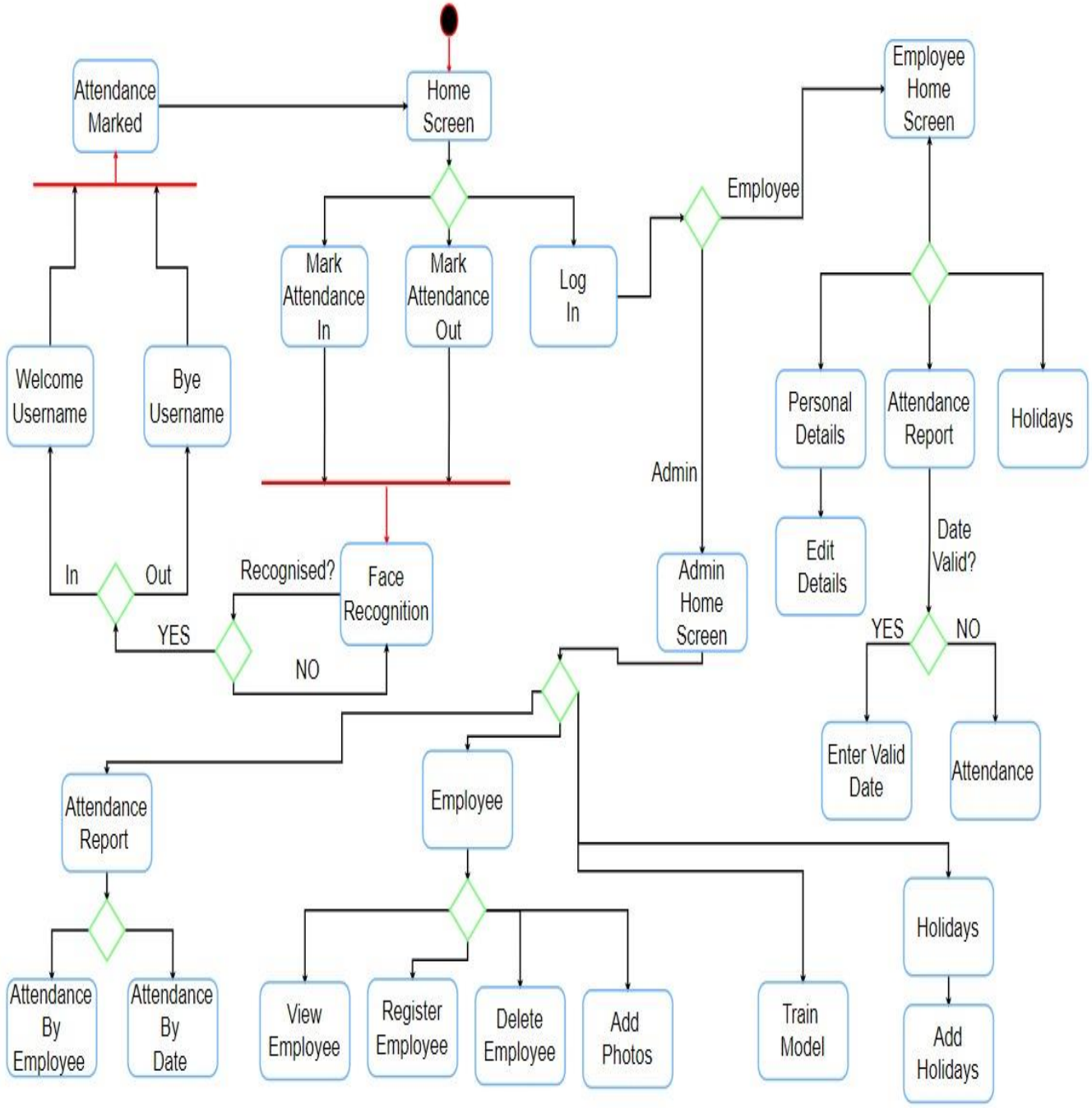


Figure 8: Activity Diagram

7.2Sequence Diagram

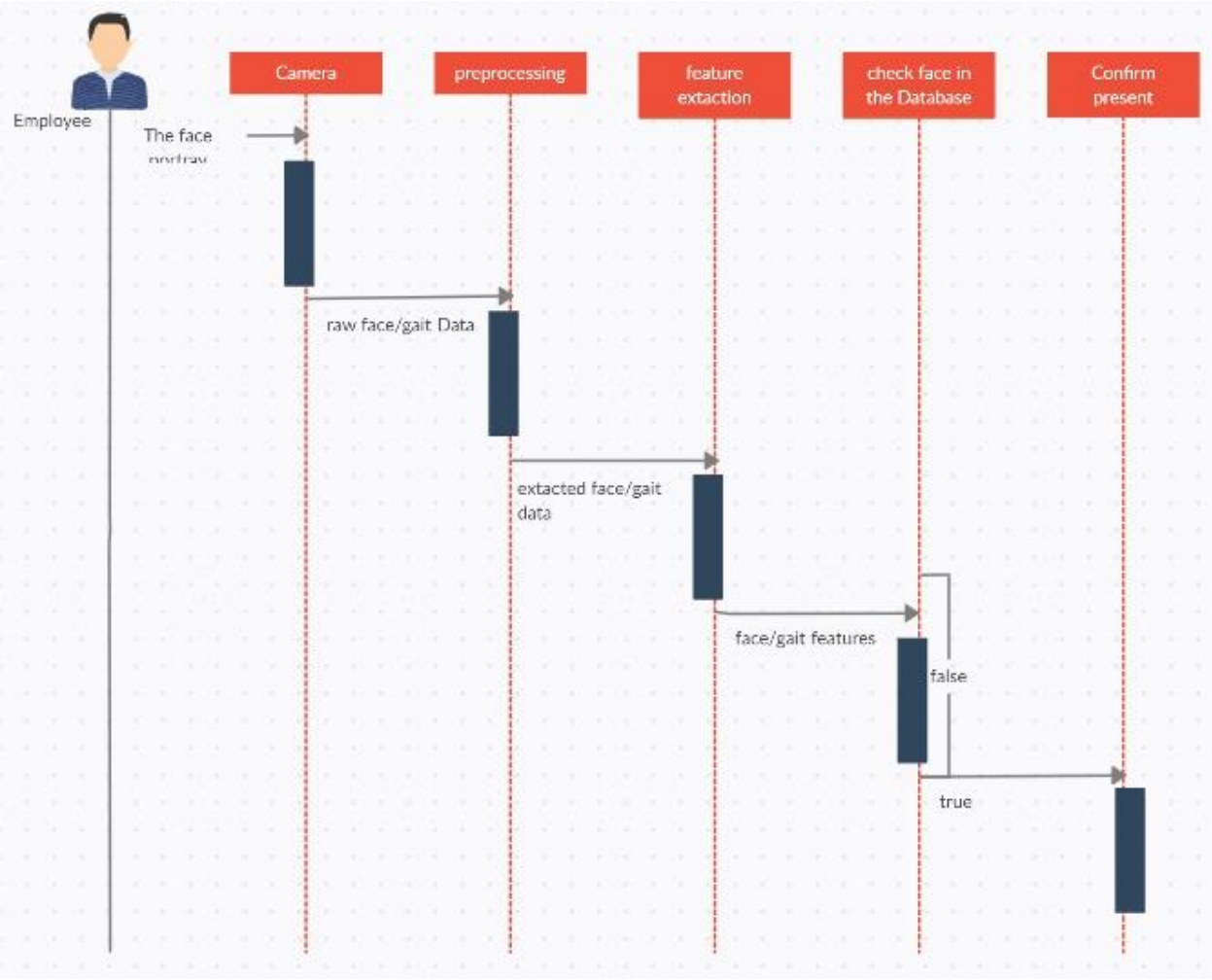


Figure 9: Sequence Diagram

7.3ER Diagram

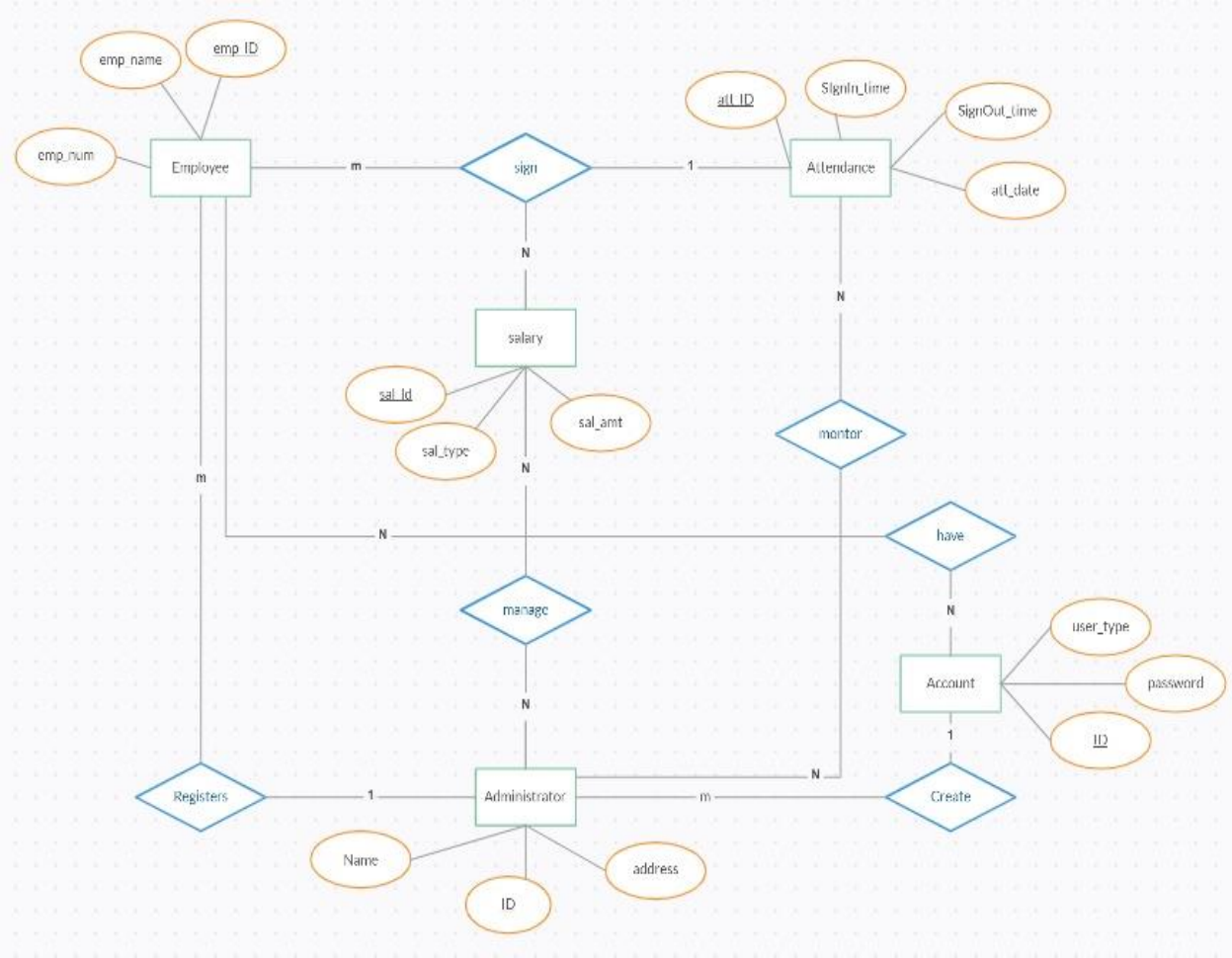


Figure 10: ER Diagram

7.4Use Case Diagram

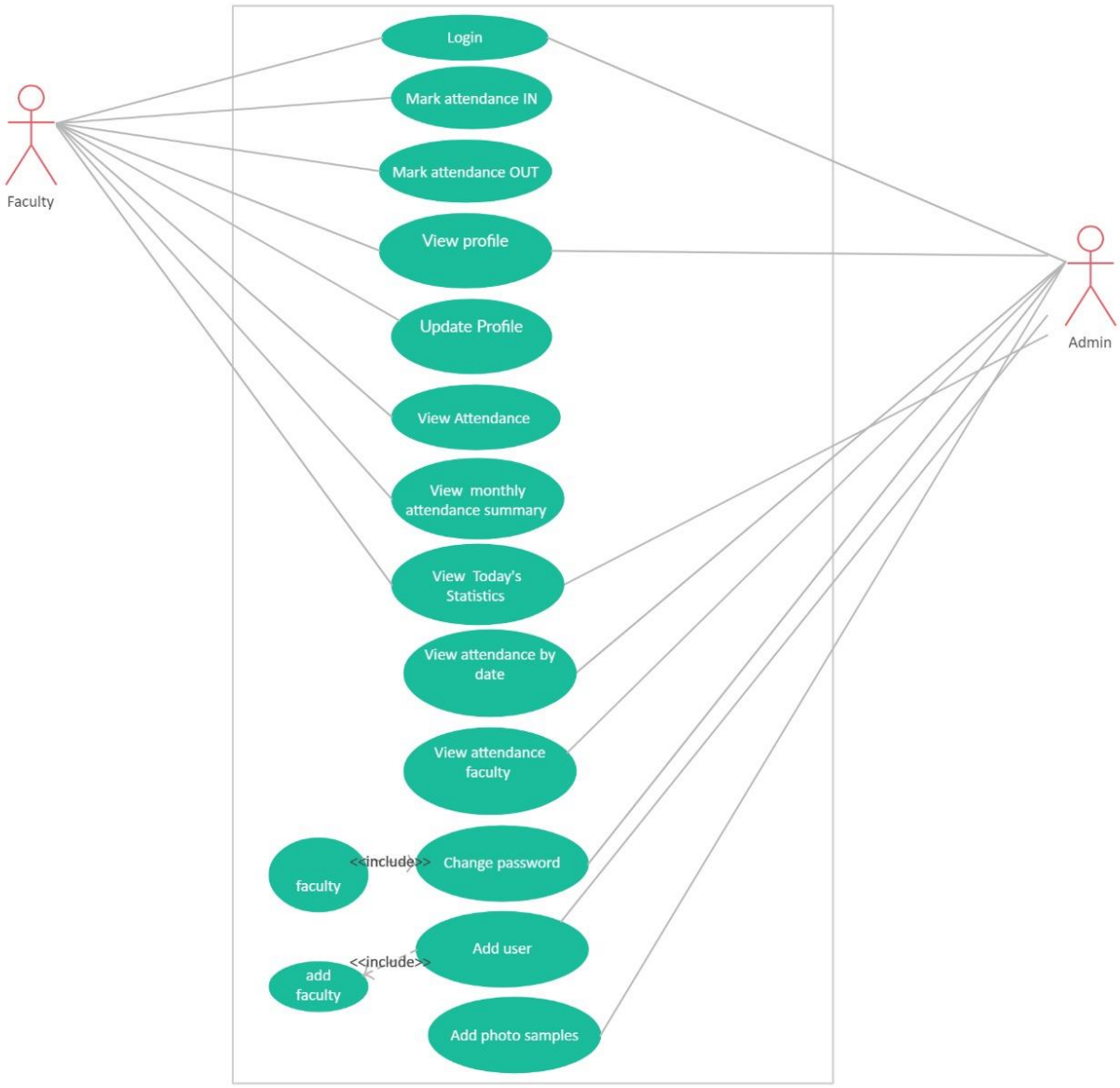


Figure 11: Use Case Diagram

Chapter 8: PROJECT PLANNING

One of the most important aspects of completing a project is project planning and approach. Project planning plays an important part in shaping the project. It is important because it addresses what is being delivered, if it is right and if it will deliver value. It also brings direction to the project. Being an important learning outcome, it is worth mentioning it in this report.

8.1 Project SRS

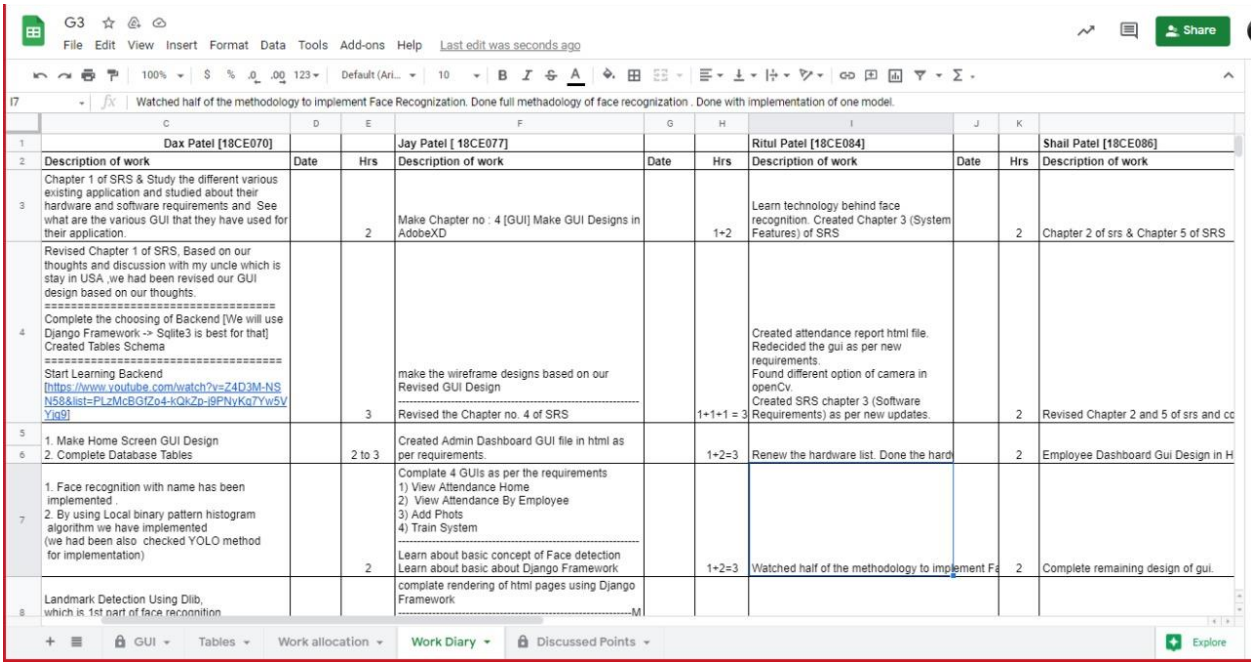
A SRS is a summary or gist of the planned project prepared in the initial period of the project timeline. It includes information such as the objective, requirements, use case descriptions, assumptions and tools and technology to be used. Writing the project SRS encouraged the team to research better upon the project area. It also pushed the team to come up with an efficient and feasible product and gave direction to the project

8.2 Weekly & Daily Planning

Weekly and daily planning have been essential in completing the project on time. The team divided the project plan into 8 weeks and the goal for each week was defined explicitly. The first day of every week was reserved for definition and division of the tasks to be completed for the week. Tools and technology to be learned were also discussed. At the end of the week, tasks completed and goals achieved were reviewed. Necessary variations were made in the plan to overcome any obstacles along the way.

Weekly plans were further divided into tasks for each day to reach each week's end goal. Tasks for each day were further divided amongst the team members. Teamwork and task division 34 has been instrumental in allowing the team to explore a wider range of libraries, models and software. A google sheet was shared with each team member and the mentor and was used to

update weekly and daily tasks along with the progress recorded, key learnings, remarks, etc. A snippet from the sheet is shown below. Figure Weekly and Daily Planning.



	Dax Patel [18CE070]			Jay Patel [18CE077]			Ritul Patel [18CE084]			Shail Patel [18CE086]
	Description of work	Date	Hrs	Description of work	Date	Hrs	Description of work	Date	Hrs	Description of work
3	Chapter 1 of SRS & Study the different various existing application and studied about their hardware and software requirements and See what are the various GUI that they have used for their application.		2	Make Chapter no : 4 [GUI] Make GUI Designs in AdobeXD		1+2	Learn technology behind face recognition. Created Chapter 3 (System Features) of SRS		2	Chapter 2 of srs & Chapter 5 of SRS
4	Revised Chapter 1 of SRS, Based on our thoughts and discussion with my uncle which is stay in USA ,we had been revised our GUI design based on our thoughts. ========== Complete the choosing of Backend [We will use Django Framework -> Sqlite3 is best for that] Created Tables Schema ========== Start Learning Backend [https://www.youtube.com/watch?v=Z4D3M-NSN58&list=PLzMcBGfZo4-kQkZp-j9PNyKq7Yw5VYjq9]		3	make the wireframe designs based on our Revised GUI Design ---------- Revised the Chapter no. 4 of SRS		1+1+1 = 3	Created attendance report html file. Redecided the gui as per new requirements. Found different option of camera in openCv. Created SRS chapter 3 (Software Requirements) as per new updates.		2	Revised Chapter 2 and 5 of srs and cc
5	1. Make Home Screen GUI Design 2. Complete Database Tables		2 to 3	Created Admin Dashboard GUI file in html as per requirements.		1+2=3	Renew the hardware list. Done the hardv		2	Employee Dashboard Gui Design in H
7	1. Face recognition with name has been implemented . 2. By using Local binary pattern histogram algorithm we have implemented (we had been also checked YOLO method for implementation)		2	Complate 4 GUIs as per the requirements 1) View Attendance Home 2) View Attendance By Employee 3) Add Phots 4) Train System ---------- Learn about basic concept of Face detection Learn about basic about Django Framework		1+2=3	Watched half of the methodology to implement Fa		2	Complete remaining design of gui.
8	Landmark Detection Using Dlib, which is 1st part of face recognition			complate rendering of html pages using Django Framework ----------M						

Figure 12: Weekly & Daily Planning in Excel

	Task Name	% Complete	Duration	Start	Finish	Pre	Resource Names
1	**Attendence System**	**95%**	**60.6 days**	**Mon 01-02-21**	**Mon 26-04-21**		
2	Formal Meeting	100%	12 days	Mon 01-02-21	Fri 19-02-21		Dax,Jay,Ronak Patel,Ritul,Shail
3	InFormal Meeting	100%	6 days	Thu 04-02-21	Wed 24-02-21	2	Dax,Jay,Shail,Ritul
4	**ANALYSIS**	**100%**	**22 days**	**Mon 08-02-21**	**Tue 09-03-21**	**3**	
5	Requirement Gathering	100%	8 days	Mon 08-02-21	Tue 02-03-21		Dax,Ritul
6	Requirement Analysis	100%	8 days	Wed 10-02-21	Tue 09-03-21	5	Jay,Shail
7	**Documentation**	**100%**	**7 days**	**Mon 15-02-21**	**Tue 23-02-21**	**4**	
8	Requirement Specification	100%	6 days	Mon 15-02-21	Tue 23-02-21		Dax,Shail,Jay,Ritul
9	**Design**	**100%**	**12 days**	**Tue 23-02-21**	**Wed 10-03-21**	**7**	
10	Wireframe Design	100%	4 days	Tue 23-02-21	Mon 01-03-21		Jay
11	UI Design	100%	4 days	Sat 27-02-21	Thu 04-03-21	10	Dax,Jay,Shail
12	Databse Design	100%	4 days	Fri 05-03-21	Wed 10-03-21	11	Dax,Jay,Ritul,Shail
13	Coding	100%	15 days	Thu 11-03-21	Wed 31-03-21	9,1	Dax,Jay,Shail,Ritul
14	Back-end	100%	12 days	Mon 29-03-21	Wed 14-04-21	13	Dax
15	Testing	60%	6 days	Wed 14-04-21	Thu 22-04-21	14	Jay,Ritul
16	Publishing	0%	1 day	Thu 22-04-21	Fri 23-04-21	15	Dax,Ritul
17	Marketing	0%	1 day	Fri 23-04-21	Mon 26-04-21	16	Shail

Figure 13: Weekly & Daily Planning in MS Project

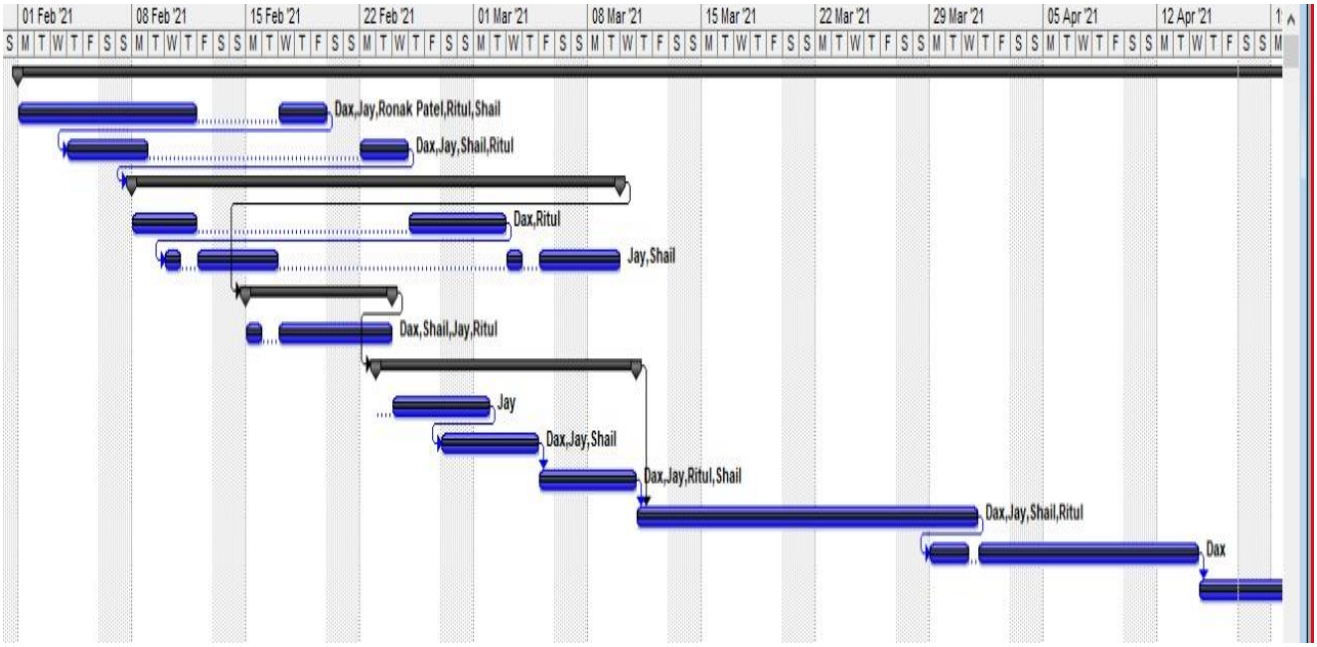


Figure 14: Gantt Chart of Weekly & Daily Planning

Chapter 9: PROJECT OUTCOME

Over a span of Four months, the team was successful in completing a proof of concept demonstrating an Attendance System based on Facial Recognition. The following screenshot of the application home page demonstrates the key functionality the application provides.

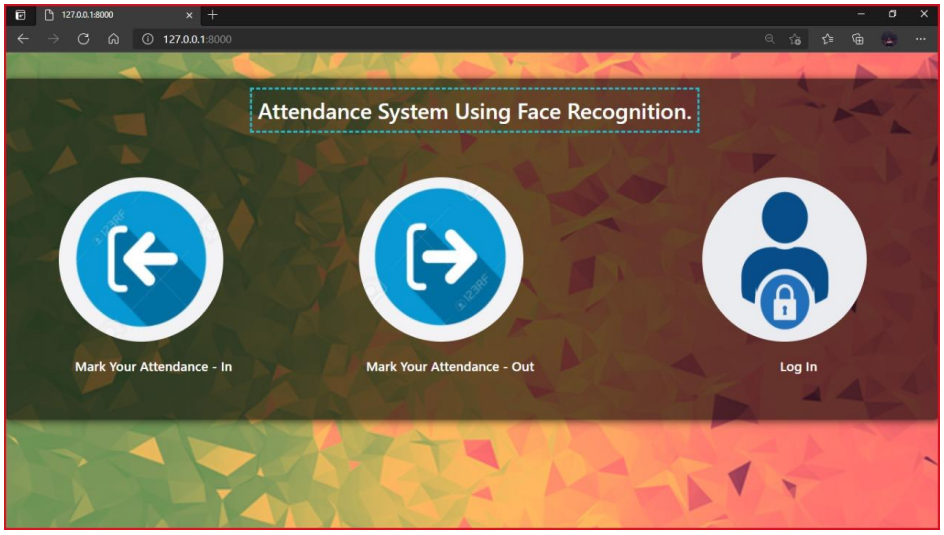


Figure 15: Home Page of the Web Application

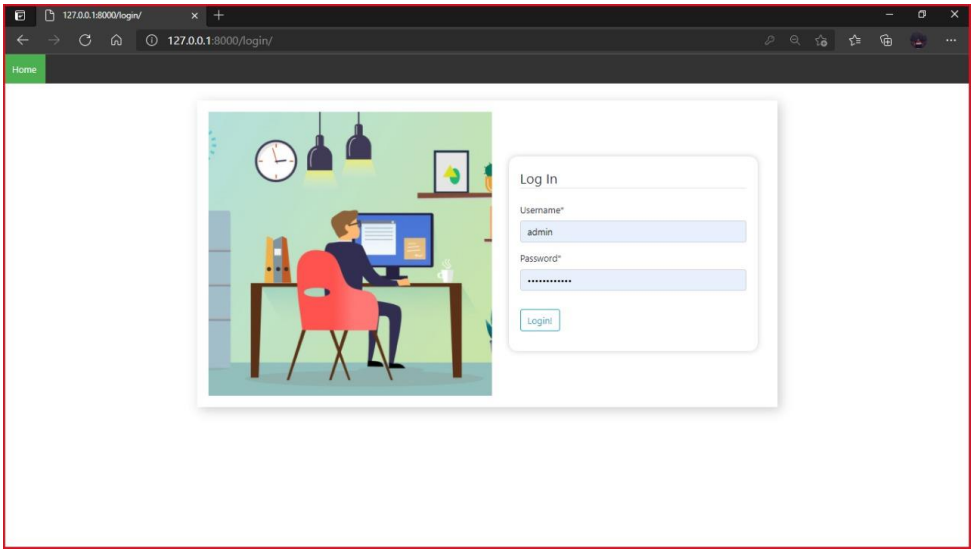


Figure 16: Login Page

9.1 About Outcome

The ‘Mark Your Attendance - In’ and ‘Mark Your Attendance - Out’ buttons provide the employee with the functionality to mark their time ins and time outs. These can be used multiple times a day and store the details of all time-ins and time-outs recorded for each employee. On clicking either of these buttons, the webcam stream is activated, and any faces present in the field of view are detected, recognized, and labeled with the corresponding name and confidence value. The attendance, time and name of the person are recorded in the database. The home page also has a login button, that supports admin and employee login. The admin dashboard is showcased in the following image.

The admin is presented with the functionality to register new employees and add their photos to the training dataset. Once photos are added, the admin can use the train button to train 38 the model, which consists of iterating through all training images, extracting their embeddings, fitting them onto a Linear SVM and saving the classifier state onto disk. The embeddings are then visualized as a scatter plot and the result is displayed on the next page.

The ‘View Attendance Reports’ button takes the employee to the attendance management section of the website. The attendance dashboard displays the attendance pattern for the previous and current week, along with the current date’s statistics. The admin is presented with another tab which allows filtering attendance by the employee and viewing it in tabular as well as graphical format. Another tab allows filtering the attendance data by date.

The employee login feature allows registered employees to log in and sees their own attendance records in tabular and graphical format.

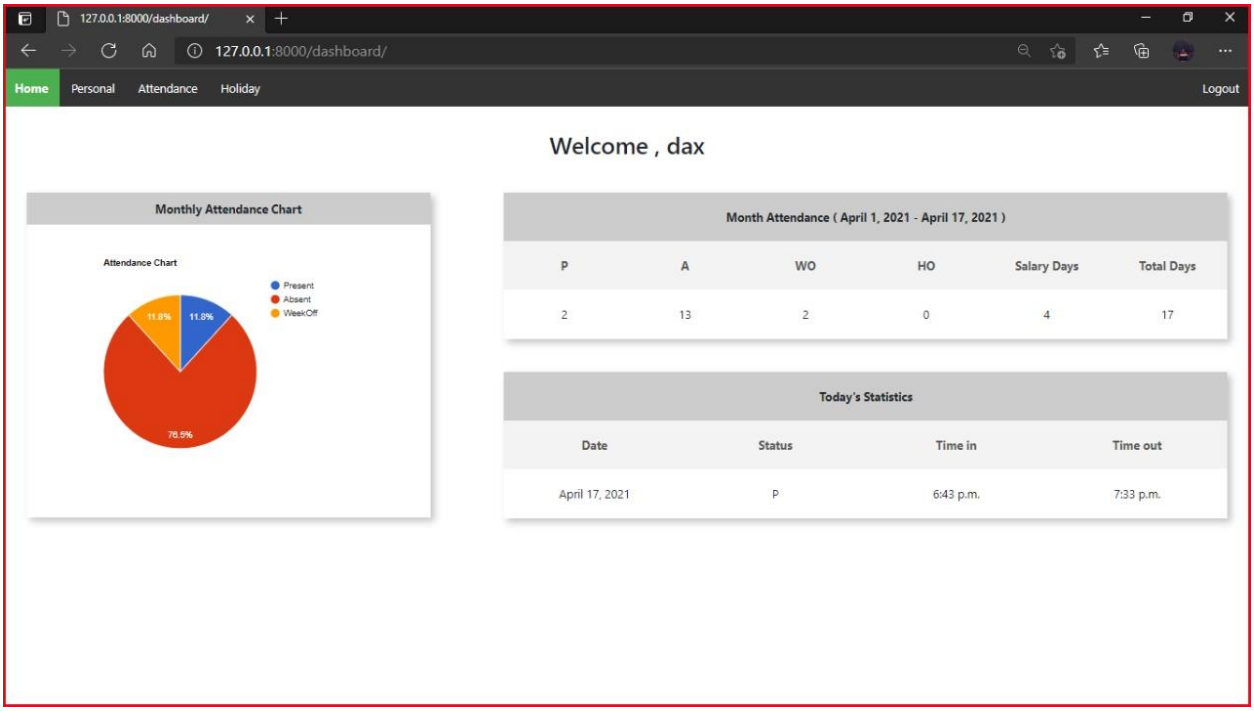


Figure 17: User Dashboard

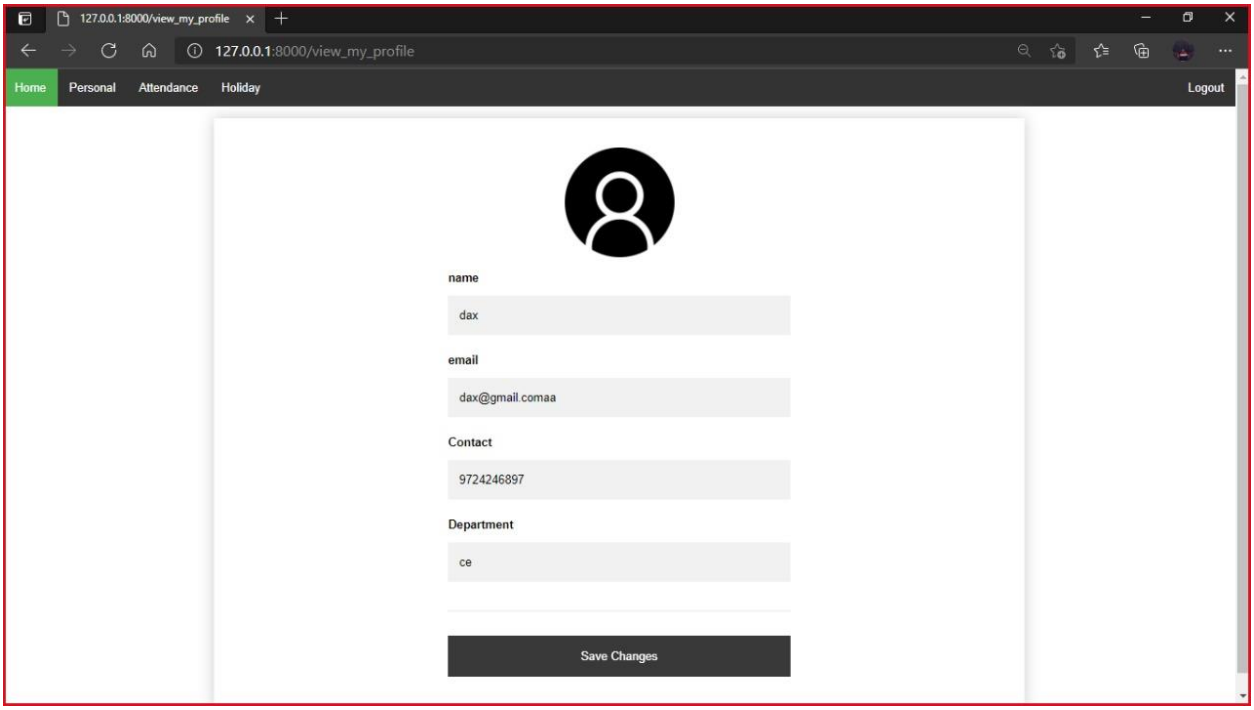


Figure 18: Edit Personal Details

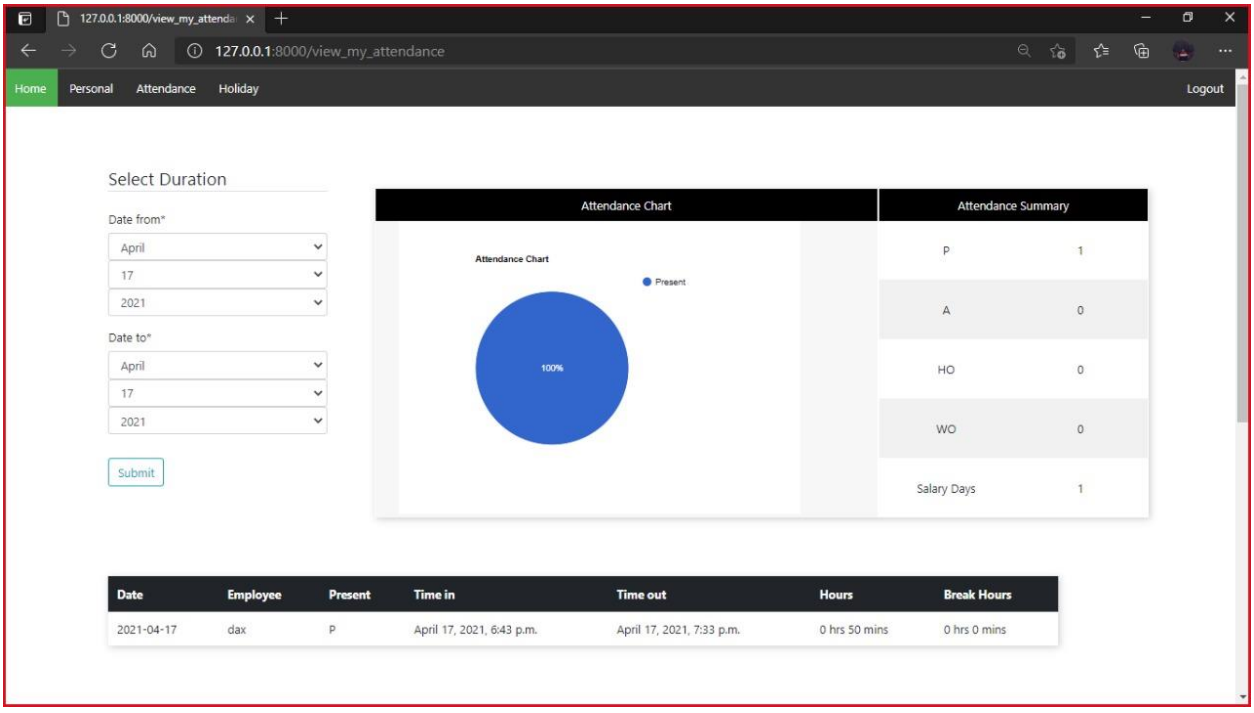


Figure 19: Specific Duration Attendance (User Side)

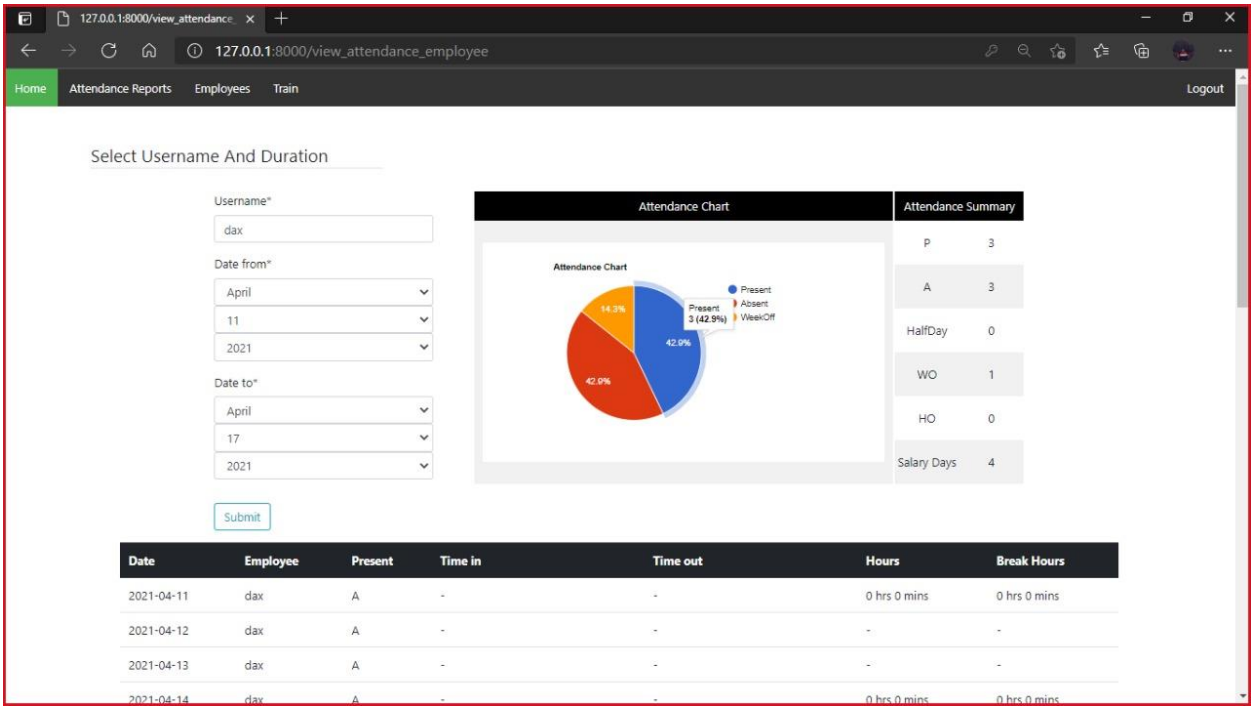


Figure 20: Specific Duration Attendance (Admin Side)

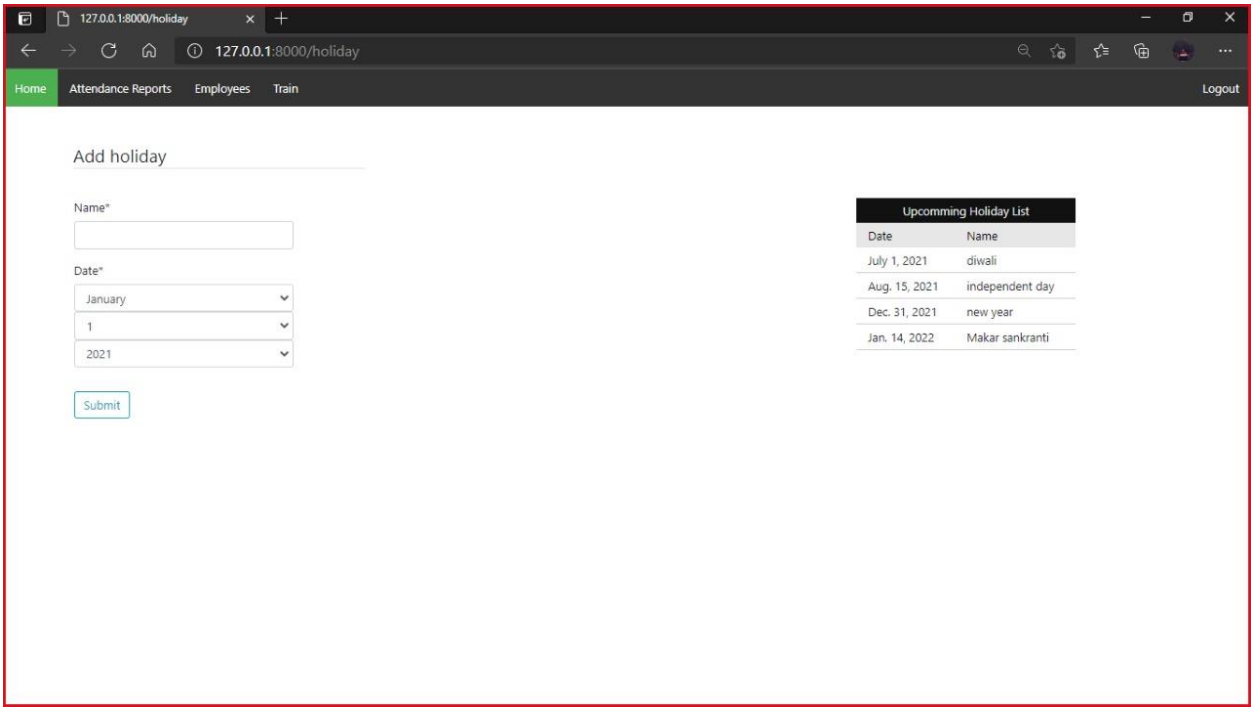


Figure 21: Holiday List & Add New Holiday

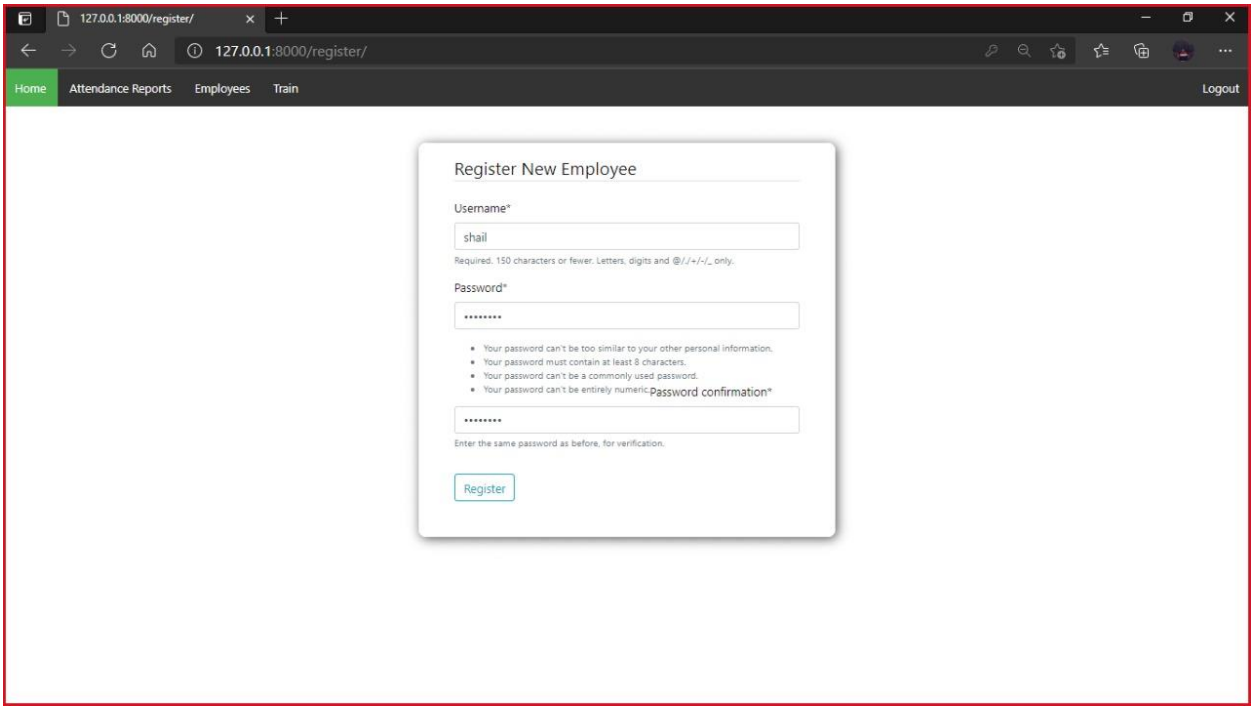


Figure 22: Register New Employee

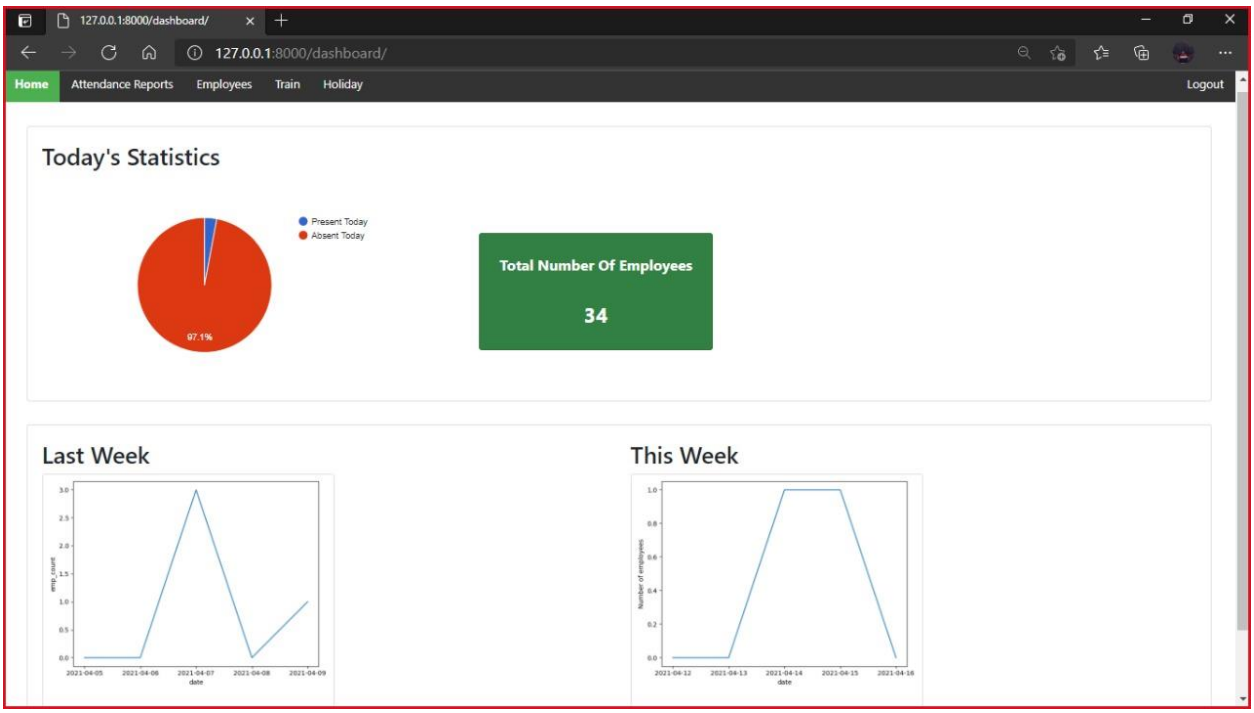


Figure 23: Today's Statistics

Figure 24: Welcome & Bye Message

Chapter 10: CONCLUSION

Faculty Attendance System has been envisioned for the purpose of reducing the errors that occur in the traditional (manual) attendance taking system. The aim is to automate and make a system that is useful our institute & also to another department as well. The efficient and accurate method of attendance in the office environment that can replace the old manual methods. This method is secure enough, reliable and available for use. No need for specialized hardware for installing the system in the campus. It can be constructed using a camera and computer.

Chapter 11: Limitations & Scope

The team identified some limitations which opens the door to opportunity for improvement in this project.

11.1 Limitations

1. The major limitation of the face recognition model is the recognition of a person's 2D image. This leads to the attendance being marked if the picture of an employee is shown. Some face recognizers are made to detect the depth of faces and hence differentiate between 3D and 2D image. The machinery used is expensive and hence cannot be incorporated in this project.

2. Another constraint is that in this project 10 images of each employee are taken for better accuracy. 10 Images per employee in a larger organization would consume a large volume to store the images.

3. The training time for our classifier takes about 20 seconds for each person. Hence for a large number of employees, it would take a very long time to train. Though training the classifier isn't something that needs to be frequently done, but it would be better if a classifier taking lesser time while maintaining the accuracy can be built.

4. The current face recognition model is 99.38% accurate

11.2 Further Scope

1. A feature which can give intruder alert can be included in the system. Furthermore, the images of unknown people can be saved in an efficient manner and displayed in the system for better security.

2. The number of training images can be reduced so that less storage is required. This can be done by removing duplicate images of the same person, or images with similar embeddings.

3. The training time can be reduced by retraining the classifier only for the newly added images.

4. A feature can be added where an employee is automatically sent a warning if his attendance or working hours are below the threshold.

5. Wrongly classified images can be added to the training dataset with the correct label so as to increase the accuracy of the recognition model

Chapter 12: REFERENCES

- **OpenCV:** opencv.org/about/
- **Django's Model:** dataflair.training/blogs/django-models/
- **Django's Admin:** https://docs.djangoproject.com/en/3.2/ref/contrib/admin/
- **Face Detection Using OpenCV:** www.superdatascience.com/blogs/opencv-face-detection
- **Video:** beta/doc/py_tutorials/py_gui/py_video_display/py_video_display.html.
- **Icons**: https://icons8.com/https://systemuicons.com/?ref=evernote.design
- **Wiki:** https://en.wikipedia.org/wiki/Face_Attendance_system