



Computer Vision Based Employee Activities Analysis

BY

MD. REZWANUL HOQUE (09201016)

NABIL TAHMIDUL KARIM (09201020)

SAIKAT LAWRENCE ROZARIO (10101007)

MD. RUMMAN BIN ASHRAF (10101028)

SUPERVISOR

MD. ZAHANGIR ALOM

Signature:

Date:

Acknowledgement

We thank all who in one way or another contributed in the completion of this thesis.

We are so grateful to the department of computer science and engineering of BRAC University. We also thank to the Professors and Lecturers of the CSE program, the librarians. We also would like to offer our gratitude to our friend, ShovonPaulinusRozario for all his efforts.

Our special and heartily thanks to our supervisor, lecturer Md. ZahangirAlom who encouraged and directed us. His challenges brought this work towards a completion. It is with his supervision that this work came into existence.

Abstract

This paper presents an automated system for human face recognition in a real time background for a company to mark the attendance of their employees. So Smart Attendance using Real Time Face Recognition is a real world solution which comes with day to day activities of handling employees. The task is very difficult as the real time background subtraction in an image is still a challenge. To detect real time human face Haar cascade is used and a simple fast Principal Component Analysis is used to recognize the faces detected with a high accuracy rate. The matched face is then used to mark attendance of the employees. Our system generates how much time each employee spends at his workstation and provides update to the employer whenever he wants. This product gives much more solutions with accurate results in user interactive manner rather than existing attendance and leave management systems.

Keywords: Real Time Face Recognition, PCA: Principle Component Analysis, NLP: Natural Language Processing, Face recognition, Haar Cascade Classifier.

Table of Contents

- 1. Introduction** 6-9
 - 1.1 Motivation 6
 - 1.2 Related works 6-8
 - 1.3 Research Methodology 8-9
 - 1.4 Outlines 9

- 2. OverallSystem Implementation** 10-41
 - 2.1 Problem definition 10
 - 2.2 Proposed Approach 12-33
 - 2.2.1 Haar Cascade Classifier..... 16-19
 - 2.2.2 Principal Component Analysis 20-25
 - 2.2.3 Database Structure..... 25-29
 - 2.2.4 Android Application..... 30-38
 - 2.2.5 Web-based Application 38-41

- 3.Results and Discussion** 42-47
 - 3.1 Discussion on Desktop Application’s Results..... 42-43
 - 3.2 Discussion on Web-based Application’s Results..... 44-47

- 4.Conclusion and Future works** 48-49
 - 4.1 Conclusion 48
 - 4.2 Future Works 49

- References** 50-51

List of tables and Figures

Figure 1: Manual Hand-written Attendance	07
Figure 2: Punch Card System	08
Figure 3: Detailed system design of face recognition	11
Figure 4: Registration Process	12
Figure 5: Employee Profile List	13
Figure 6: Employee Profile Deletion Process.....	14
Figure 7: Common Haar Features.....	16
Figure 8: Summed are of integral image integral image	18
Figure 9: Summed area of rotated	18
Figure 10: Cascaded Classifier	19
Figure 11: ER Diagram for Profile	26
Figure 12: ER Diagram for Record	27
Figure 13: Dynamic Time-stamp Table Generation	28
Figure 14: ER Diagram for Recognition	29
Figure 15: ER Diagram for Recognition.....	29
Figure 16: HTTP Request and Response between application and server	31
Figure 17: Login View of the android application	33
Figure 18: Employee View of the Profile	37
Figure 19: Admin View of the Profile	37
Figure 20: Homepage View of the website	39
Figure 21: Employee Profile	40
Figure 22: View of the employee list	41
Figure 23: Face Detection	42
Figure 24: Face Recognition	43
Figure 25: User going offline	43
Figure 26: Graphical Representation of the employee activities.....	45
Figure 27: Tabular Form of the employee activities	46
Figure 28: Tabular Form of the employee activities in android app	47

1. Introduction

In our entire job market there are a lot of people holding a desk job, which means they have a fixed workstation in a small or a large working space. Using close circuit cameras we can detect the faces in the frame and count the time of being in the frame. As because the price of close circuit camera is a bit high, primarily we will be using a webcam to implement our system. The system will be detecting the faces and show rectangular boxes around the faces inside the frame. Simultaneously it will be counting the time of each face in staying live in the frame.

1.1 Motivation

When it comes to information security in a corporate network, internal threats dominate over external ones, so it is important not only to monitor possible leakage by means of data interception and analysis, but also to control network activities of the staff. The lack of comprehensive control over business processes may lead to an increased number of potential insiders and disloyal employees, which often entails a serious deterioration of the company's performance.

1.2 Related works

Person identification is one of the most crucial building blocks for smart interactions. Among the person identification methods, face recognition is known to be the most natural ones, since the face modality is the modality that uses to identify people in everyday lives. Although there are other methods such as,

1. Manual Hand-written Attendance

In many institutions where people mainly hold a desk job still have manual attendance system where employees register their entry time and departure time manually in an attendance sheet. Even in a technically advanced world, a vast majority of companies are still using a time tracking system that relies on handwritten paper time cards. This traditional payroll process requires people to manually count the number of hours for each day, each pay period, and each employee. Collecting time cards from multiple job sites, trying to decipher illegible handwritten cards, and calculating employee work hours isn't a simple process, especially in the construction industry. It can be a slow system that's prone to error and a real time-waster for business owners, job foremen, or payroll personnel.



Figure 1: Manual Hand-written Attendance

2. Punch Card System

One of the most popular biometric time clock attendance systems is punch card attendance system. A punch card is a flat and stiff paper with notches cut in it and contains digital information. In punch card attendance system, employees use this punch or proximity card for in and/or out. To use a punch card, employees just need to wave the card near a reader, which then ensures if the

correct person is logging in and/or out. Punch card attendance system records the accurate in and/or out time of each employee.



Figure 2: Punch Card System

Disadvantage of 2nd system: They are slow to load. The data density is low, they are susceptible to being damaged by water and anything else that destroys paper, they can get accidentally shuffled, they take lots of storage space, nobody has card readers anymore.

1.3 Research Methodology

Basically this research is aimed for implementing a system that is capable of identifying the employees in an organization marking their attendance. Therefore face recognition is used to mark the attendance of the employees. Smart Attendance using Real Time Face Recognition (SMART-FR) provides flexibility to identify several employees at the same time separately rather than identifying one by one. To increase the accuracy, efficiency and reliability of the recognition, algorithms are needed. Principle Component Analysis (PCA) and Haar cascade are used to address those tasks. This system has a standalone application, web-based application and an android application. Standalone application deals with the face recognition process and the attendance marking

process. Web based application mainly deals with the employee information and display the employee attendance sheet. The android application serves the same purpose as the website. All applications are linked to a centralized database.

1.4 Outlines

This thesis is organized in four chapters including this introduction. Chapter 2 attempts to find answer to the first question proposing and investigating the entire algorithm used here step by step to give a shape of our proposed system employee activity analysis where video feeds are retrieved from camera and an input image is taken which is processed to detect face. Chapter 3 is concerned about experimental results of this system and its discussions in detail. Chapter 4 summarizes the main contributions of this thesis. Finally, further research directions are suggested.

2. Overall System Implementation

2.1 Problem definition

Real time video has been captured using A4tech 5 mega pixel webcam and it has been processed into frames. These frames have been used as input images for the system. Haar Cascade Classifier, one of the most popular methods in the face detection process has been used to detect faces in these input images. Later on, Principal Component Analysis (PCA) has been used to recognize the detected faces in these input images and afterwards, the attendance marking process starts and each employee has his records get filled up in the database every time the detection algorithm is run.

In order to make the system more efficient and faultless, we have identified the need to run the recognition algorithm twice and the detection algorithm to run in a loop. Therefore, the more cameras we install, the more complicated the system becomes in terms of running the detection algorithm. As we are running the detection algorithm in a loop, therefore, the more cameras we install the longer the loop gets and each camera has to wait that much longer to run the detection algorithm again to check if there is a face in front of the desk. When an employee leaves his chair, it results in the detection process not finding any faces. At that time, the employee's database is again updated by showing the employee was present for that duration of time.

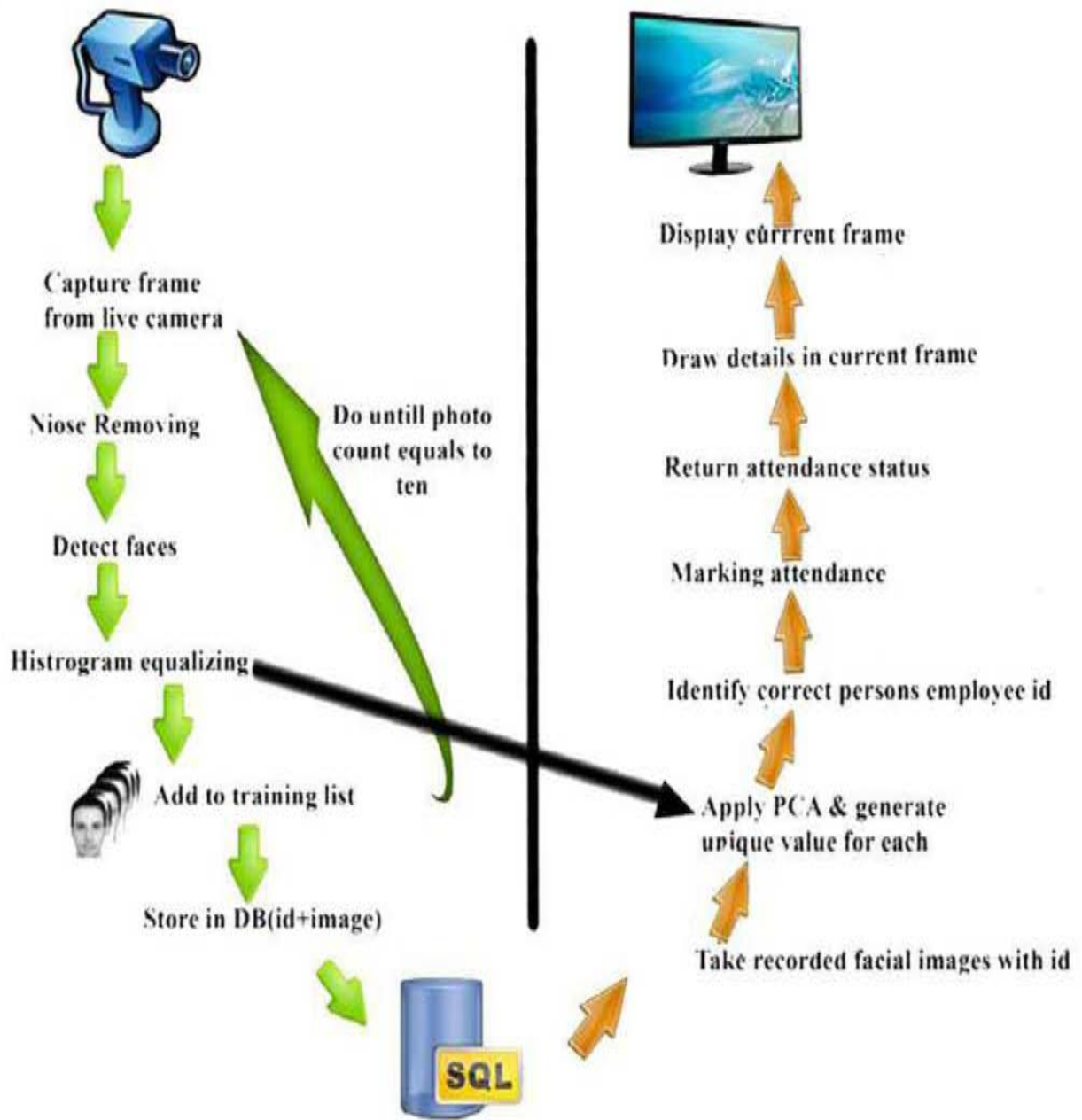


Figure 3: Detailed system design of face recognition

2.2 Proposed Approach

First of all we will implement a desktop application which will be programmed for the employees as well as the employers to register themselves under the software package we are providing. The user will fill up some basic fields in the desktop application. Fields are given below:

- **First Name**
- **Last Name**
- **Address**
- **Email**
- **Mobile Number**
- **Landline Number**
- **Job Title**
- **Department Name**
- **Picture (Upload)**
- **Password**

The screenshot shows a desktop application window titled "Full Information". At the top, there is a header bar with the text "Full Information". Below the header, there is a sub-header "Add New Employee". The form contains the following fields:

Employee ID	<input type="text"/>	Email	<input type="text"/>
First Name	<input type="text"/>	Mobile Number	<input type="text"/>
Last Name	<input type="text"/>	Land Number	<input type="text"/>
Address	<input type="text"/>		
Job Title	<input type="text"/>	Department	<input type="text"/>
Hire Date	<input type="text"/>	Picture Path	<input type="text"/>
Password	<input type="text"/>	CameraDeskID	<input type="text"/>

At the bottom of the form, there are two icons: a file upload icon and a red 'X' cancellation icon.

Figure 4: Registration Process

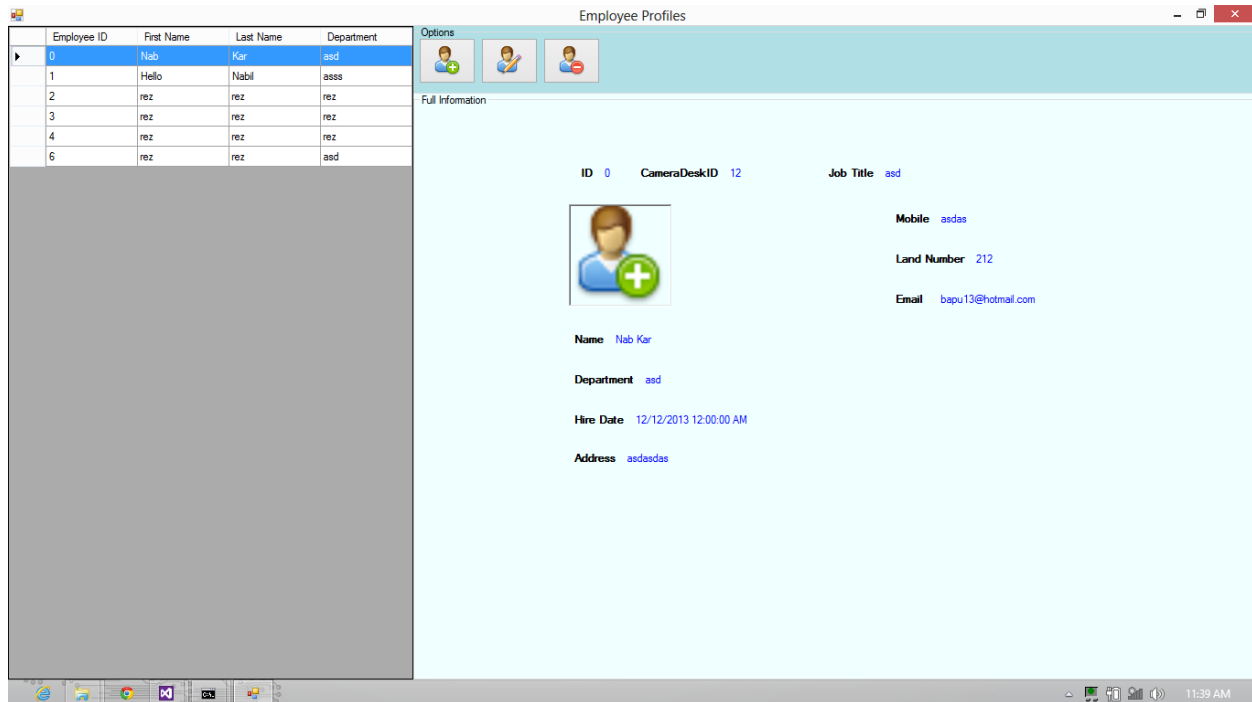


Figure 5: Employee Profile List

After submitting the information required, the registration request will be checked by an admin who will grant the request as registered. When the request is granted an employee profile will be created to the database. The admin will fill up some confidential data field for each user.

- **Employee ID**
- **Camera ID**
- **Hire Date**

Each employee who has already registered can see their profile and working details in a website. Firstly they have to log in to the website with the EmployeeID as “Username” and the password. After logging in, they can see their profile and can also edit the profile to some extent. For example: they cannot change the fields which were added by the admin. Admin can look over all the employee profiles and their working activities. Similarly the registered employees can have an android application if they use cell

phone with android operating system. The employer himself can act as admin over here. S/he can retrieve all information about the employees whereas the employees can only see the personal details of themselves and working details.

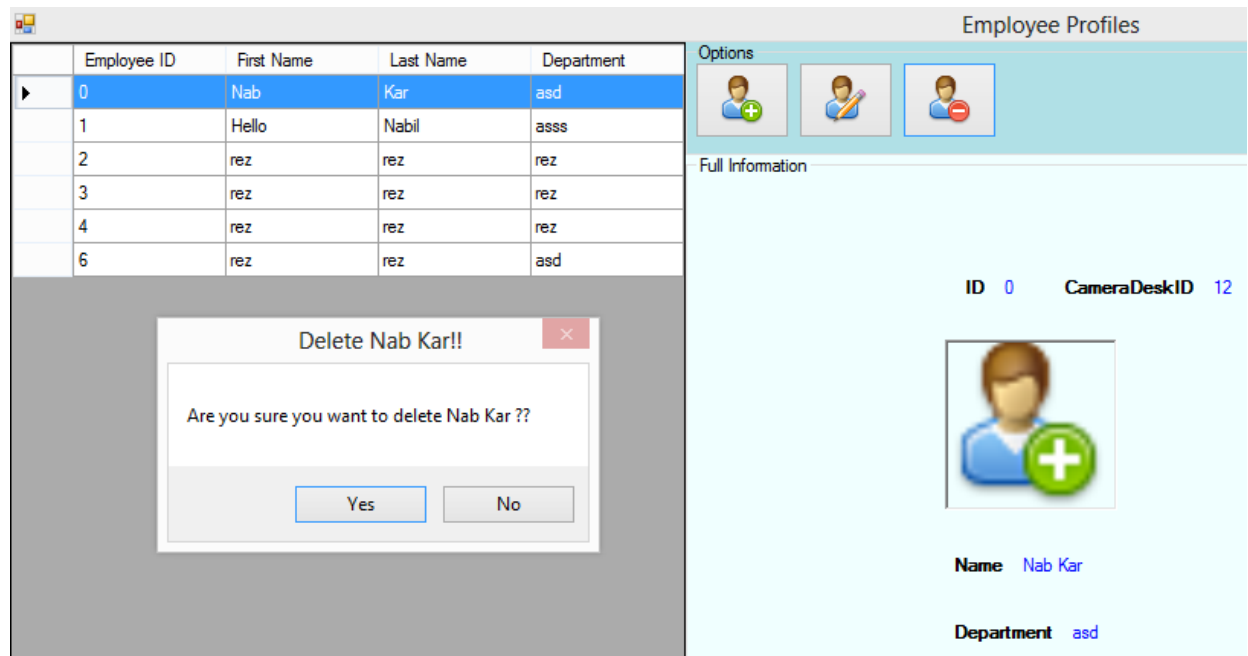


Figure 6: Employee Profile Deletion Process

First of all we have converted the input image which is in RGB to GRAY image because Haar Cascade Classifier works efficiently for gray images rather than for RGB images. This smoothen the picture for getting clear edges to detect the key points which are in scale-space extreme step. Then Haar Cascade Classifier is run to detect faces in the gray images. Detecting human face require that Haar classifier cascades first be trained. In order to train the classifiers, this PCA algorithm and Haar feature algorithms must be implemented. Later on, the PCA algorithm is run to recognize the detected faces in the gray image and start the attendance marking process. Whenever an employee's face has been recognized at the office main entrance, a timestamp has been recorded under "entry time" of that employee's table in the "timetable" database and five minutes time is added with the entry time and recorded under the "exit time" of

that employee's table in the "timetable" database. This step is done assuming it takes 5 minutes to reach the employee's desk from the office main entrance.

After entering his designated office, another A4tech 5 mega pixel webcam is used to run the same face detection and recognition process. However, this time the recognition process will run only one time to make the system is more efficient. After the employee has sat on his chair, Haar Cascade Classifier is run to detect whether there is a human face or not on the chair. Afterwards, PCA algorithm is run on the detected faces to check if the desired employee has sat on his allocated chair or if someone else is sitting on that employee's chair. In our proposed system, each employee has a "CameraDeskID" which essentially identifies which employee is sitting on which desk and under which camera. After running the recognition algorithm, our proposed system will take a time-stamp and add that record under the "start_entry" of the "active" table in the internal local database. At that same time, the corresponding "CameraDeskID" of that employee will be recorded under the "CameraDeskID" of the "active" table in the internal local database. When an employee decides to leave his chair and at the same time when the Haar Cascade Classifier is run, it will not find any face at that time. If the detection algorithm does not find any face for 15 times at a stretch, then another timestamp will be taken at that point and the difference between the second time-stamp and the first time-stamp is actually the amount of time the employee spent at his desk. Both the time-stamps will be recorded under the "start_entry" and "exit_entry" respectively of that employee's table in the "timetable" database and the difference will be shown graphically in the web-based application and numerically in the mobile-based application. After sending the data to the "time-table" database, the "start-entry" field and the "counter" field used to count up to 15 times of the internal local database, will be reset for that "CameraDeskID".

The detection algorithm will continue to run in a loop and when the employee gets back to his desk, the face detection algorithm will eventually find a face. At that moment, PCA the face recognition algorithm will be run to make sure the desired employee has sat on his allocated desk not someone else. If the desired employee's face is recognized, a

time-stamp will be taken to count his office timing again and will be recorded in the local database. Later on, if the employee decides to leave his chair, like before, again the face detection algorithm will run and if it does not find any face for 15 times at a stretch, another time-stamp will be taken and similarly the difference between the second time-stamp and the first time-stamp will be shown in the on the website as well as on the android application and both the time-stamps will be recorded under the “start_entry” and “exit_entry” respectively of that employee’s table in the “timetable” database. In this way, the process will continue to go on for the entire day till the office is open.

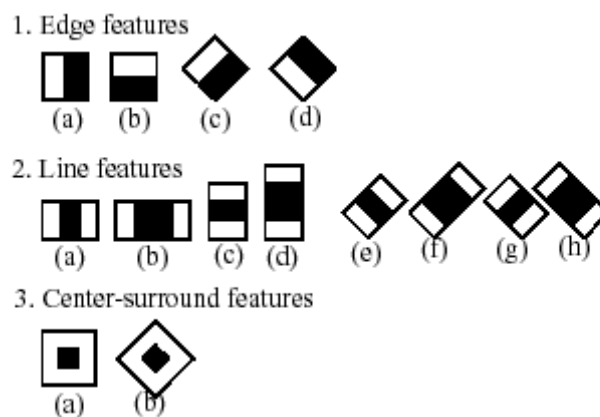


Figure 7: Common Haar Features

2.2.1 Haar Cascade Classifier

The core basis for Haar classifier object detection is the Haar-like features. These features, rather than using the intensity values of a pixel, use the change in contrast values between adjacent rectangular groups of pixels. The contrast variances between the pixel groups are used to determine relative light and dark areas. Two or three adjacent groups with a relative contrast variance form a Haar-like feature. Haar-like features, as shown in Figure 4 are used to detect an image. Haar features can easily be scaled by increasing or decreasing the size of the pixel group being examined. This allows features to be used to detect objects of various sizes.

Integral Image

The simple rectangular features of an image are calculated using an intermediate representation of an image, called the integral image. The integral image is an array containing the sums of the pixels' intensity values located directly to the left of a pixel and directly above the pixel at location (x, y) inclusive. So if $A[x,y]$ is the original image and $AI[x,y]$ is the integral image then the integral image is computed as shown in equation 1 and illustrated in Figure 5.

$$AI[x,y] = \sum_{x' \leq x, y' \leq y} A(x',y') \quad (1)$$

The rotated integral image is calculated by finding the sum of the pixels' intensity values that are located at a forty five degree angle to the left and above for the x value and below for the y value. So if $A[x,y]$ is the original image and $AR[x,y]$ is the rotated integral image then the integral image is computed as shown in equation 2 and illustrated in Figure 6.

$$AR[x,y] = \sum_{x' \leq x, x' \leq x - |y - y'|} A(x',y') \quad (2)$$

It only takes two passes to compute both integral image arrays, one for each array. Using the appropriate integral image and taking the difference between six to eight array elements forming two or three connected rectangles, a feature of any scale can be computed. Thus calculating a feature is extremely fast and efficient. It also means calculating features of various sizes requires the same effort as a feature of only two or three pixels. The detection of various sizes of the same object requires the same

amount of effort and time as objects of similar sizes since scaling requires no additional effort.

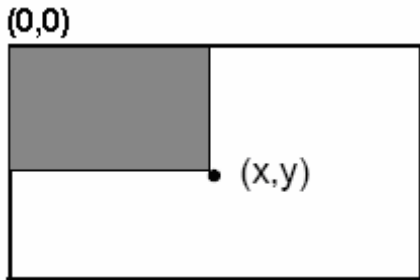


Figure 8: Summed area of integral image

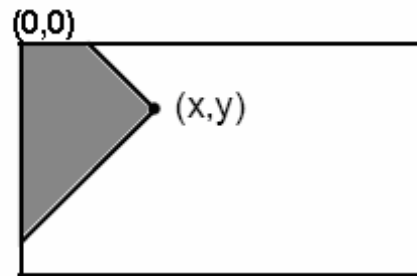


Figure 9: Summed area of rotated image

Classifiers Cascaded

Although calculating a feature is extremely efficient and fast, calculating all 180,000 features contained within a 24×24 sub-image is impractical. Fortunately, only a tiny fraction of those features are needed to determine if a sub-image potentially contains the desired object. In order to eliminate as many sub-images as possible, only a few of the features that define an object are used when analyzing sub-images. The goal is to eliminate a substantial amount, around 50%, of the sub-images that do not contain the object. This process continues, increasing the number of features used to analyze the sub-image at each stage.

The cascading of the classifiers allows only the sub-images with the highest probability to be analyzed for all Haar-features that distinguish an object. It also allows one to vary

the accuracy of a classifier. One can increase both the false alarm rate and positive hit rate by decreasing the number of stages. The inverse of this is also true.

The basic principle of the Viola-Jones face detection algorithm is to scan the detector many times through the same image –each time with a new size. Even if an image should contain one or more faces it is obvious that an excessive large amount of the evaluated sub-windows would still be negatives (non-faces). This realization leads to a different formulation of the problem: Instead of finding faces, the algorithm should discard non-faces. Hence the need for a cascaded classifier arises.

The cascaded classifier is composed of stages each containing a strong classifier. The job of each stage is to determine whether a given sub-window is definitely not a face or maybe a face. When a sub-window is classified to be a non-face by a given stage it is immediately discarded. Conversely a sub-window classified as a maybe face is passed on to the next stage in the cascade. It follows that the more stages a given sub window passes, the higher the chance the sub-window actually contains a face.

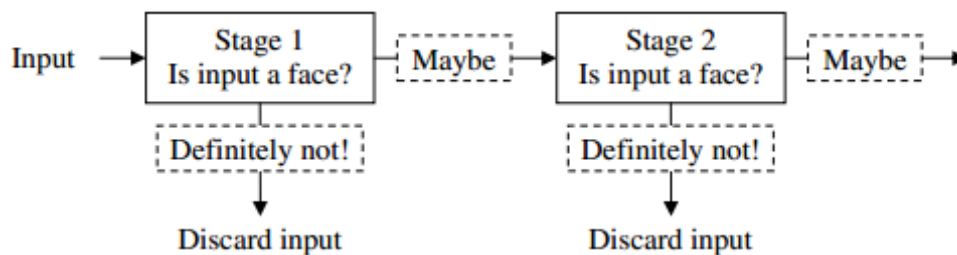


Figure 10: Cascaded Classifier

In a single stage classifier one would normally accept false negatives in order to reduce the false positive rate. However, for the first stage in the staged classifier false positives are not considered to be a problem since the succeeding stages are expected to sort them out. Therefore Viola-Jones prescribes the acceptance of many false positives in the initial stages.

2.2.2 Principal Component Analysis(PCA)

The Principal Component Analysis (PCA) is one of the most successful techniques that have been used in image recognition and compression. PCA is a statistical method under the broad title of *factor analysis*. The purpose of PCA is to reduce the large dimensionality of the data space (observed variables) to the smaller intrinsic dimensionality of feature space (independent variables), which are needed to describe the data economically. This is the case when there is a strong correlation between observed variables.

The jobs which PCA can do are prediction, redundancy removal, feature extraction, data compression, etc. Because PCA is a classical technique which can do something in the linear domain, applications having linear models are suitable, such as signal processing, image processing, system and control theory, communications, etc.

Face recognition has many applicable areas. Moreover, it can be categorized into face identification, face classification, or sex determination. The most useful applications contain crowd surveillance, video content indexing, personal identification (ex. driver's license), mug shots matching, entrance security, etc. The main idea of using PCA for face recognition is to express the large 1-D vector of pixels constructed from 2-D facial image into the compact principal components of the feature space. This can be called eigenspace projection. Eigenspace is calculated by identifying the eigenvectors of the covariance matrix derived from a set of facial images (vectors).

Mathematics of PCA

A 2-D facial image can be represented as 1-D vector by concatenating each row (or column) into a long thin vector. Let's suppose we have M vectors of size N ($=$ rows of image \times columns of image) representing a set of sampled images. p_j 's represent the pixel values.

$$x_i = [p_1 \dots p_N]^T, i = 1, \dots, M(1)$$

The images are mean centered by subtracting the mean image from each image vector. Let m represent the mean image.

$$m = \frac{1}{M} \sum_{i=1}^M x_i \quad (2)$$

And let w_i be defined as mean centered image,

$$w_i = x_i - m \quad (3)$$

Our goal is to find a set of e_i 's which have the largest possible projection onto each of the w_i 's. We wish to find a set of M orthonormal vectors e_i for which the quantity

$$\lambda_i = \frac{1}{M} \sum_{n=1}^M (e_i^T w_n)^2 \quad (4)$$

is maximized with the orthonormality constraint

$$e_i^T e_k = \delta_{ik} \quad (5)$$

It has been shown that the e_i 's and λ_i 's are given by the eigenvectors and eigenvalues of the covariance matrix

$$C = WW^T \quad (6)$$

Where W is a matrix composed of the column vectors w_i placed side by side. The size of C is $N \times N$ which could be enormous. For example, images of size 64×64 create the covariance matrix of size 4096×4096 . It is not practical to solve for the eigenvectors of C directly. A common theorem in linear algebra states that the vectors e_i and scalars λ_i can be obtained by solving for the eigenvectors and eigenvalues of the $M \times M$ matrix $W^T W$. Let d_i and μ_i be the eigenvectors and eigenvalues of $W^T W$, respectively.

$$W^T W d_i = \mu_i d_i \quad (7)$$

By multiplying left to both sides by W ,

$$W^T W (W d_i) = \mu_i (W d_i) \quad (8)$$

which means that the first $M-1$ eigenvectors e_i and eigenvalues λ_i of $W W^T$ are given by $W d_i$ and μ_i respectively. $W d_i$ needs to be normalized in order to be equal to e_i . Since we only sum up a finite number of image vectors, M , the rank of the covariance matrix cannot exceed $M-1$ (The -1 come from the subtraction of the mean vector m).

The eigenvectors corresponding to nonzero eigenvalues of the covariance matrix produce an orthonormal basis for the subspace within which most image data can be represented with a small amount of error. The eigenvectors are sorted from high to low according to their corresponding eigenvalues. The eigenvector associated with the largest eigenvalue is one that reflects the greatest variance in the image. That is, the smallest eigenvalue is associated with the eigenvector that finds the least variance. They decrease in exponential fashion, meaning that the roughly 90% of the total variance is contained in the first 5% to 10% of the dimensions.

A facial image can be projected onto M' ($\ll M$) dimensions by computing

$$\Omega = [v_1 v_2 \dots v_{M'}]^T \quad (9)$$

Where $v_i = e_i^T w_i$. v_i is the i^{th} coordinate of the facial image in the new space, which came to be the principal component. The vectors e_i are also images, so called, *eigenimages*, or *eigenfaces* in our case. They can be viewed as images and indeed look like faces.

So, Ω describes the contribution of each eigenface in representing the facial image by treating the eigenfaces as a basis set for facial images. The simplest method for determining which face class provides the best description of an input facial image is to find the face class k that minimizes the Euclidean distance

$$\epsilon_k = \|\Omega - \Omega_k\| \quad (10)$$

Where Ω_k is a vector describing the k^{th} face class. If ϵ_k is less than some predefined threshold θ_ϵ , a face is classified as belonging to the class k .

Face recognition

Once the eigenfaces have been computed, several types of decision can be made depending on the application. What we call face recognition is a broad term which may be further specified to one of the following tasks:

- *identification* where the labels of individuals must be obtained,
- *recognition* of a person, where it must be decided if the individual has already been seen,
- *categorization* where the face must be assigned to a certain class.

PCA computes the basis of a space which is represented by its training vectors. These basis vectors, actually eigenvectors, computed by PCA are in the direction of the largest

variance of the training vectors. As it has been said earlier, we call them eigenfaces. Each eigenface can be viewed as a feature. When a particular face is projected onto the face space, its vector describes the importance of each of those features in the face. The face is expressed in the face space by its eigenface coefficients (or weights). We can handle a large input vector, facial image, only by taking its small weight vector in the face space. This means that we can reconstruct the original face with some error, since the dimensionality of the image space is much larger than that of face space. Each face in the training set is transformed into the face space and its components are stored in memory. The face space has to be populated with these known faces. An input face is given to the system, and then it is projected onto the face space. The system computes its distance from all the stored faces.

However, two issues should be carefully considered:

1. What if the image presented to the system is not a face?
2. What if the face presented to the system has not already been learned, i.e., not stored as a known face?

The first defect is easily avoided since the first eigenface is a good face filter which can test whether each image is highly correlated with itself. The images with a low correlation can be rejected. Or these two issues are altogether addressed by categorizing following four different regions:

1. Near face space and near stored face => known faces
2. Near face space but not near a known face => unknown faces
3. Distant from face space and near a face class => non-faces
4. Distant from face space and not near a known class => non-faces

Since a face is well represented by the face space, its reconstruction should be similar to the original; hence the reconstruction error will be small. Non-face images will have a large reconstruction error which is larger than some threshold θ_r . The distance ϵ_k determines whether the input face is near a known face.

2.2.3 Database Structure

For the entire project we have used MYSQL database. MYSQL is the world's most widely used open-source relational database management system (RDBMS). It runs as a server which provides multi-user access to a number of databases. The SQL phrase stands for Structured Query Language. The purpose of using MYSQL database in our project is it is very popular for web applications and it has a highly user friendly graphical user interface.

There will be two types of database in our project. One is for the desktop application and another one is for server side data management. For the desktop application the database will be internal which will contain the id of each employee and their respective pictures for the recognition part we will execute. On the other hand the server side database will contain two types of database. First one will contain a table which stores basic information for each employee and their working details and also their profile picture. Second one will be the kind which stores each employee's arrival and leaving time to and from his desk; it will also note his arrival and departure from office.

The database for the employee profile will have a table that contains all the basic information about the employee. In this table 14 fields.

1. **EmployeeID**
2. **FirstName**
3. **LastName**
4. **Address**
5. **Email**
6. **MobileNumber**
7. **LandlineNumber**
8. **JobTitle**
9. **DepartmentName**

- 10. HireDate
- 11. Picture
- 12. Password
- 13. CameraDeskID
- 14. ProfileType

Among this 14 fields EmployeeID, HireDate and CameraDeskID will be given by an admin who will certainly provide permissions to access the software and the website as well as the android application.

This table will be populated when an employee registers and the admin accepts that. Each employee will have a unique EmployeeID. It will be the primary key for the Employee entity. ProfileType will decide if the user is an admin or not. Every employee will have a CameraDeskID. It means that under which camera the employee will be working on his desk.

The ER Diagram of the Database Profile is given below:

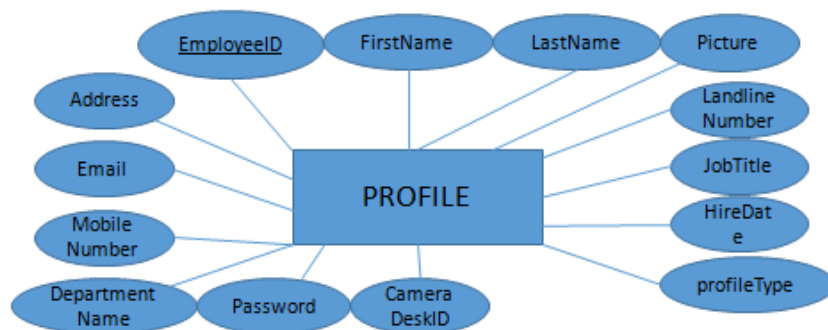


Figure 11: ER Diagram for Profile

The second type of table that contains timings for each employee will have maximum of 62 and minimum of 56 fields. For example: 1s, 1e, 2s, 2e,, 31s, 32e . This database will be generated dynamically. In the starting of every month our program will calculate the days in that month and create a table for each employee with daysinmonth

*2 number of fields. Every two fields will be related to each other. 1s and 1e mean in the first day of month starting time of work and ending time of work for an employee. In this database entry time, departure time and the time spent behind the desk will be stored for every registered employee.

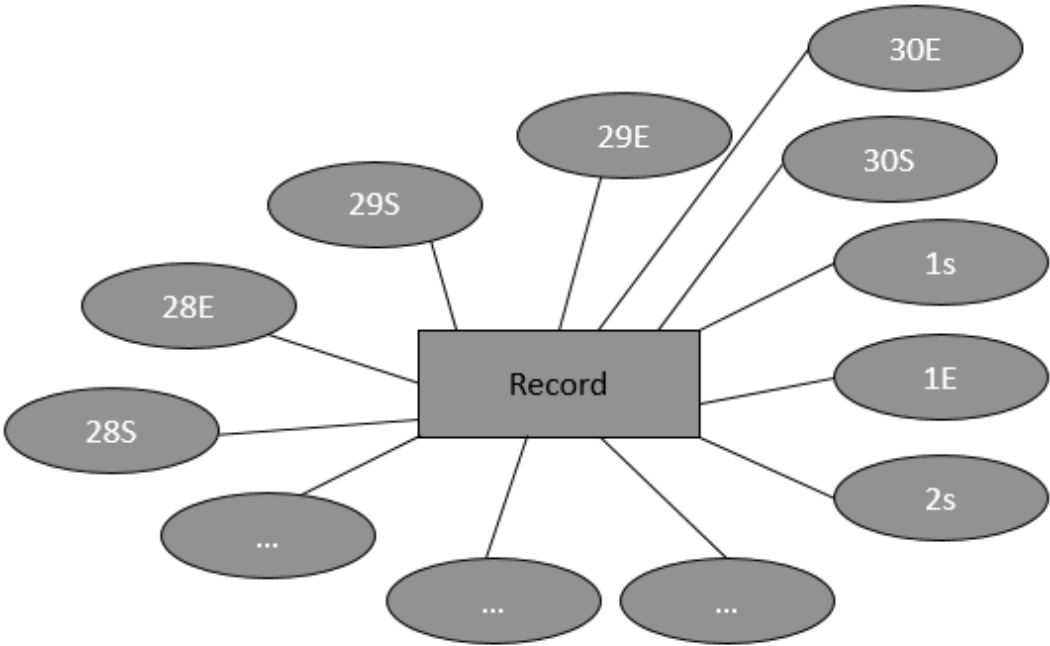
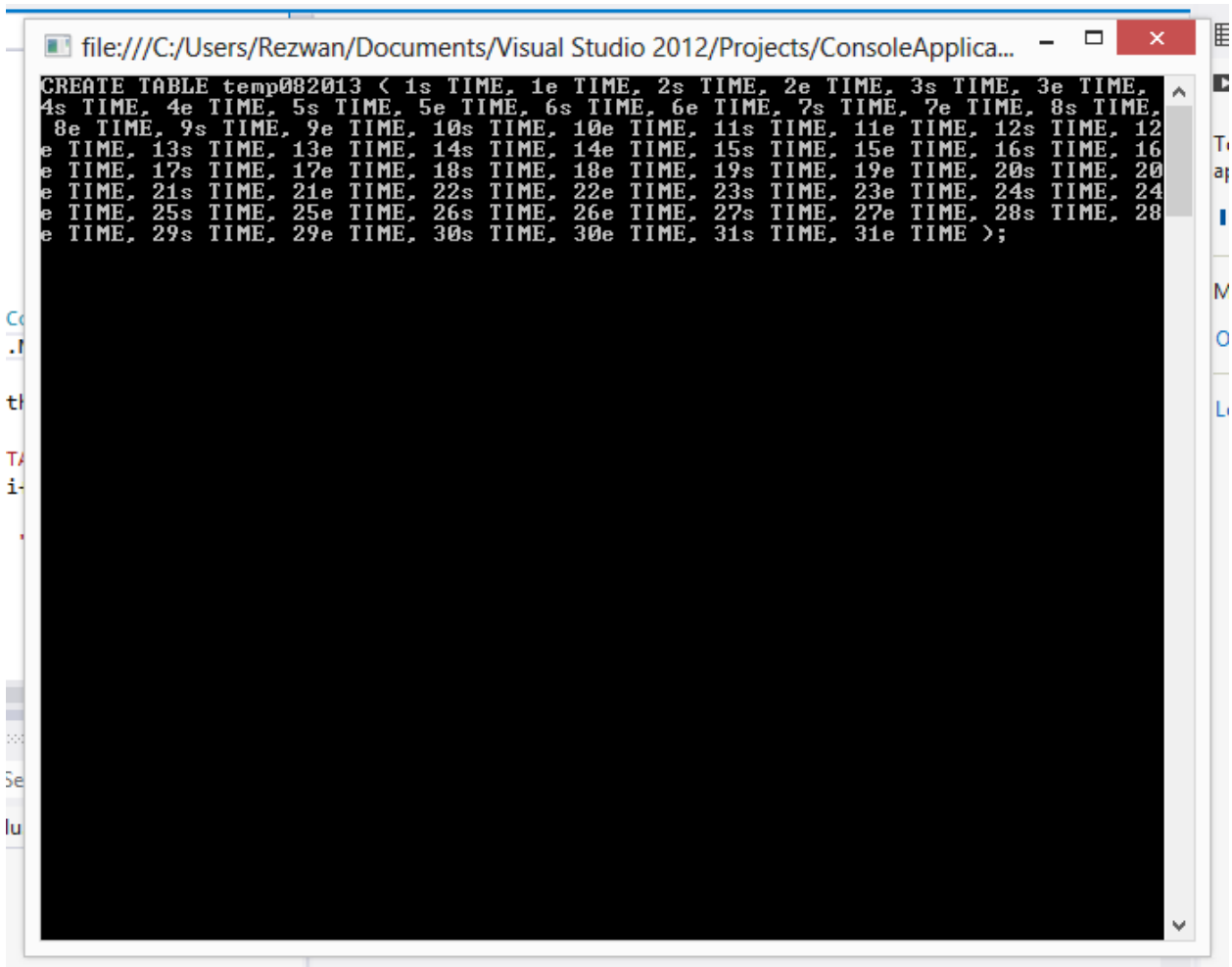


Figure 12: ER Diagram for Record

Each starting time will have an end time. If there is an entry for 1s, there must be an entry for 1e. Every day when an employee enters the office the camera will recognize him and put a start time in the database which means that the employee has arrived to the office. After he gets to his desk another start time will take entry to generate his working time. When the employee moves away from his desk the end time will be noted and the duration between start time and end time will be calculated. The next time he arrives to the desk a new start time entry will be given.



```
file:///C:/Users/Rezwan/Documents/Visual Studio 2012/Projects/ConsoleApplica...
CREATE TABLE temp082013 < 1s TIME, 1e TIME, 2s TIME, 2e TIME, 3s TIME, 3e TIME,
4s TIME, 4e TIME, 5s TIME, 5e TIME, 6s TIME, 6e TIME, 7s TIME, 7e TIME, 8s TIME,
8e TIME, 9s TIME, 9e TIME, 10s TIME, 10e TIME, 11s TIME, 11e TIME, 12s TIME, 12
e TIME, 13s TIME, 13e TIME, 14s TIME, 14e TIME, 15s TIME, 15e TIME, 16s TIME, 16
e TIME, 17s TIME, 17e TIME, 18s TIME, 18e TIME, 19s TIME, 19e TIME, 20s TIME, 20
e TIME, 21s TIME, 21e TIME, 22s TIME, 22e TIME, 23s TIME, 23e TIME, 24s TIME, 24
e TIME, 25s TIME, 25e TIME, 26s TIME, 26e TIME, 27s TIME, 27e TIME, 28s TIME, 28
e TIME, 29s TIME, 29e TIME, 30s TIME, 30e TIME, 31s TIME, 31e TIME >;
```

Figure 13: Dynamic Time-stamp Table Generation

In this table every two fields are reserved for a day. For the first day of the month 1s and 1e are allocated. All the entries in the first day will be in 1s and 1e. Similarly in the 25th day of the month all the entries will take place in 25s and 25e. Every month a new table will be created for each employee to keep records of the working hour.

The internal local database stores the CameraDeskID, start_time and a counter in the “active” table and the two co-ordinates of the Rectangle and the CameraDeskID in the “area” table. Each camera has a unique id. In the office, each desk will also have a unique DeskID. CameraID and DeskID jointly create the CameraDeskID. An employee will be working on a particular desk which will be covered by a particular camera. For

every employee in the frame the camera will generate two co-ordinate points of the Rectangle.

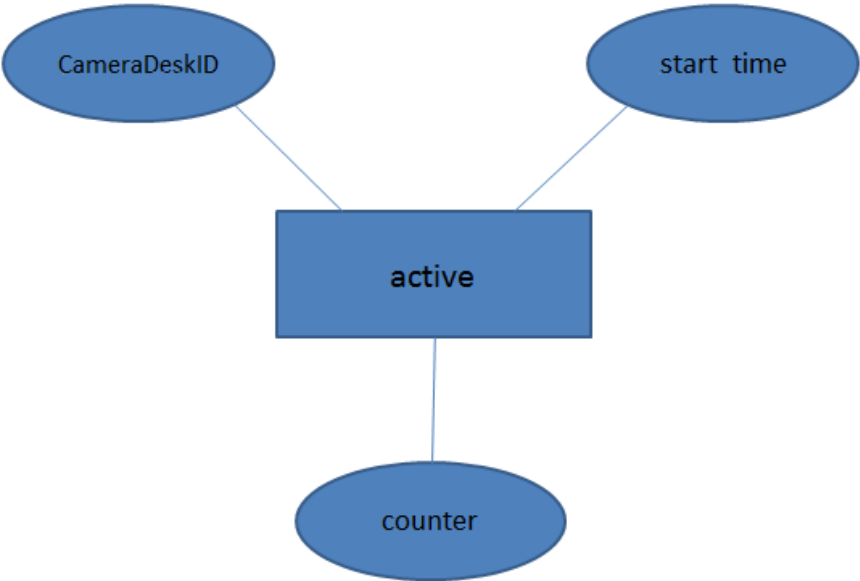


Figure 14: ER Diagram for Recognition

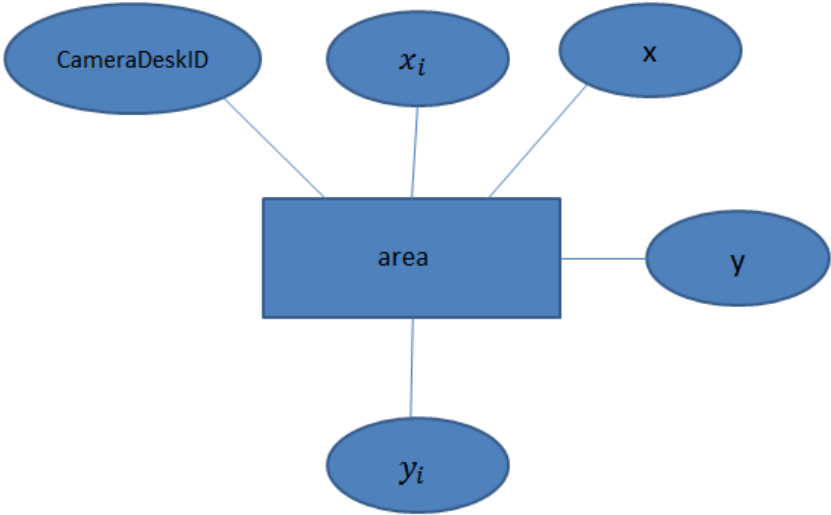


Figure 15: ER Diagram for Recognition

2.2.4 Android Application

In order to provide the mobility of the software to the employer, we have also developed an android application. The reason behind choosing android platform is it is the mostly used and preferred platform in the world. Android's increasing popularity has led us to believe that it will be used more and more in future, therefore we have developed our application in Android.

The Android application will be used by both the employer and the employees. This application will show the data of employees' activity according to their access privilege. Which means using this application, when an administrator logs in, he/she will be able to view his own activity log which will show his own data of attendance in workspace and also of the employees who he administrates. However, when an employee logs in he will be able to view only his own data and statistics of attendance and presence in workspace.

In order to explain how the application works, the structure of an android application must be explained first. In every android project there are some basic elements in each package where a developer can create and modify his/her code in order for that application to work. These elements are in different folders and files. They are:

1. src
2. gen
3. Android version
4. assets
5. AndroidManifest.xml

Among these a developer is allowed to add files to **src** and **assets** folder. All the java files are added to the src folder and the layouts (designs for application) and strings used are added in the assets folder. He can modify the AndroidManifest.xml file for his application to work properly.

Our Android application needs to connect and communicate with the internet for the use of it. When this application needs any data it sends an HTTP Request. Using internet

when this request reaches the server, the server provides appropriate HTTP Response and sends it back.

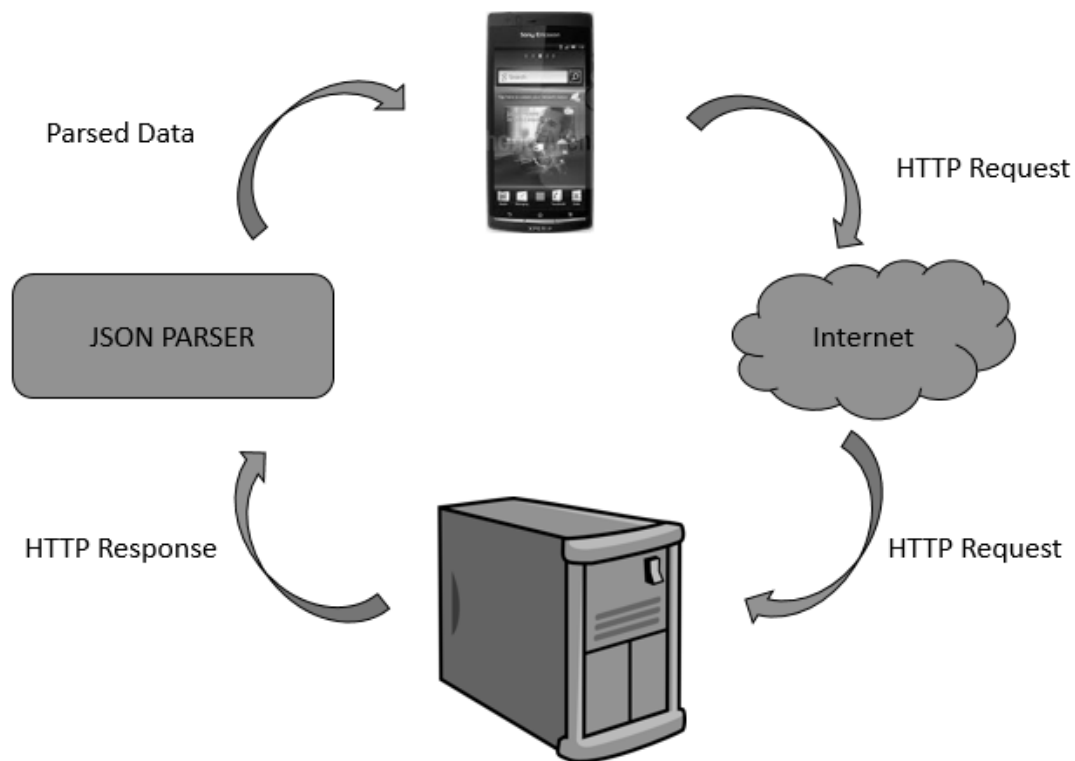


Figure 16: HTTP Request and Response between application and server.

We use a simple JSON Parser to parse this response and make it readable for the application. After receiving the Parsed HTTP response, the application uses this data and shows appropriate representation of it according to the user. If it's an admin it shows him all the data and if it's a normal user it will show him only his data.

A brief description of our android application will be given below. In our android application, the src folder contains the default package. In this package there are four JAVA files.

1. Login.java
2. JSONParser.java
3. ReadProfiles.java
4. ShowDetailedActivity.java

Login.java

The Login.java class is used for the authentication of a user. This class asks for a username and a password from the user and then verifies it. After taking the username and password this class cross-references these data with the data in the mysql database. If they match this class allows the login for that user.

This class contains:

1. onCreate method
2. onClick method
3. AttemptLogin class

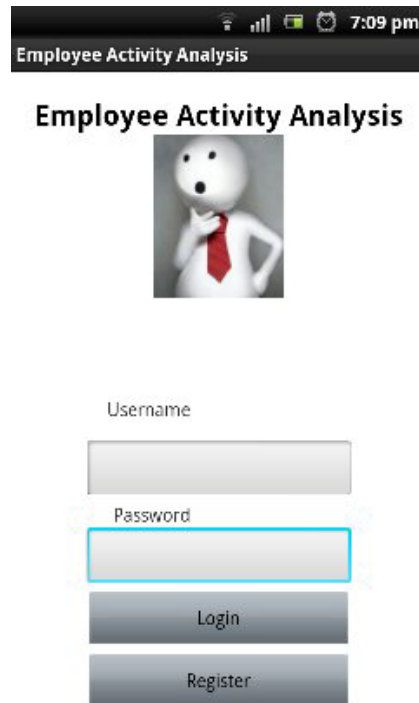


Figure 17: Login View of the android application

Oncreate Method

Oncreate is a method in this class. In any android class this onCreate method works like a main method in a java class.

This method first indicates which layout is chosen for this class for a user to view using **setContentView** method. Then it creates two **EditText** objects **user** and **pass** and also points to the textfields in the layout, which is represented by these objects.

Then this method creates a **Button** object **msubmit** and also points to the **button** in the layout, which is represented by this object. Lastly, it enables the register listeners for this button.

onClick Method

onclick is also a method in **Login** class. When a button is clicked which belongs to this class this method gets called. If login button is clicked, this method calls another method **execute** in **AttemptLogin** class.

AttemptLogin class

This a class inside the login class. This class control what the application will do before the HTTP request is send, while it is being send and after it is sent. It contains **onPreExecute**, **doInBackground** and **onPostExecute** methods to control do these functionalities. The onPreExecute method creates a message for the user and shows it before our request is executed. The method **doInBackground** is most important among these three methods.

This method takes the **username** and **password** variable and creates name-value pair using a List for using it as parameter while creating JSON object. After that, it creates a JSON object with our websites login url, our method POST and the parameters. Upon successful login it saves the user data using **SharedPreferences**. Then we create a **Bundle** to pass our username and password to the next class ReadProfiles.java for further usage of these data.

The **onPostExecute** method gets called after the **doInBackground** method finishes executing. This method in our project dismisses the message which was being shown from **onPreExecute**. Then it shows the message such as, "Login Successful" or "Invalid Credentials" according to the success or failure of a user to login.

JSONParser.java

This class is used for the purpose of making an HTTP request, storing an HTTP response and after that parsing that response. For accomplishing these task it uses the POST and GET methods.

This class contains:

1. getJSONFromUrl method
2. makeHttpRequest method

getJSONFromUrl

This method reads each line from a url and then parses it to JSON objects. Firstly, this class takes a url as a parameter. It creates a DefaultHttpClient and HttpPost object, then it creates anHttpResponse executing HttpPost on this DefaultHttpClient. After that it extracts the data from the response using HttpEntity and later on it opens an InputStream with contents received from the response.

In order to parse the InputStream help of BufferedReader is needed. BufferedReader object is created with the help of InputStream object. In order to make parsing more easier a StringBuilder object is also created.

A loop is used to read the data line by line and then each line is added to the StringBuilder Object. Later on the StringBuilder object is converted to a String. A JSON object is created from the String and then this object is returned from this method.

makeHttpRequest

makeHttpRequestmethod is used to make HTTP requests to the website of our project for getting appropriate data. This method takes in three parameters, the url where it will make the HTTP request, the type of request method (POST or GET) and a name-value pair. Firstly, this method checks for the type of method.

If it's a POST method then, It creates a DefaultHttpClient and HttpPost object. This method also creates aUrlEncodedFormEntity object from the name-value parameters and hands this as an entity to the HttpPost. It creates an HttpResponse executing HttpPost on the DefaultHttpClient. After that data is extracted from the HTTP response.

If the method is GET then just like before a DefaultHttpClient object is created. In order to encode urls in an organized way we format our parameter so that it is suitable for use in a GET method and store it in a String. This String is then added to the url. An HttpGet Object is created from this url. Then it creates an HttpResponse executing HttpGet on this DefaultHttpClient. After that it extracts the data from the response using HttpEntity and later on it opens an InputStream with contents received from the response. Rest of the parsing is exactly same as the parsing in the **getJSONFromUrl method**. At the end the JSON object is returned.

ReadProfiles.java

This class works basically to show the information about an employee. When any user logs in this class, first it checks whether the user is an admin or not. If the user is an admin, this class reveals information about all employees to the admin. However if the user is not an admin which means he/she is an ordinary employee this class only fetches the information about that particular user and shows it.

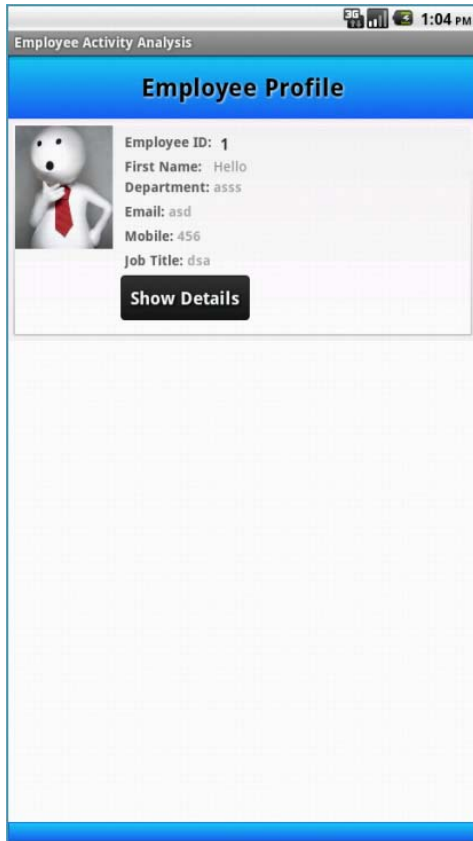


Figure 18: Employee View of the profile

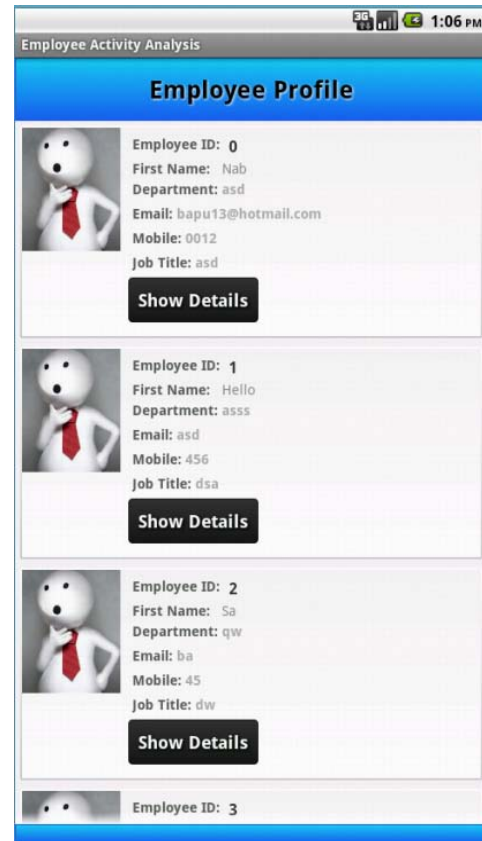


Figure 19: Admin View of the profile

This class contains:

1. onCreate Method
2. onResume Method

onCreate

In this method the application only selects the layout which will be used to show all the fetched data.

onResume

onResume method calls the methods in **LoadComments** class which does the further job. **LoadComments** creates an environment before showing all the expected results

and finally reveals that. In this class we have another three methods. The first one is **onPreExecute** method. Here it just shows a screen of loading and rest of the work is done in background. The next method is **doInBackground** which calls **updateJSONData**.

updateJSONData creates an array list and fetches the assigned data from the php script. From the login page the username and password were passed to this class. Using this username and password this method checks from the array list if the username and password matches with the admin or not. If it matches then **updateJSONData** method inserts information about all employee to the array list and store. If the username and password does not matches the method only gets the information of that employee stores them.

Finally in the method **onPostExecute** we dismiss the loading page and show the information using **updateList** method. Here we create a ListAdapter named adapter which takes five parameters. First one is the context. Second one is a list. In this case the array list which contains all the information. We get that from the method **updateJSONData**. Thirdly it's the layout where each list of information will be viewed. The fourth one is attribute names in our database. These are stored in here as an array of string. Finally it is the array of integer which contains the chronological value for the attribute name to be shown in the layout.

2.2.5 Web-based Application

Our proposed system includes a website for both the employees and the employer. From this website, the employer can keep track of his employees' activities as well as know about the basic informations of the employees. The website also provides the employer to see the employees in an organized list.

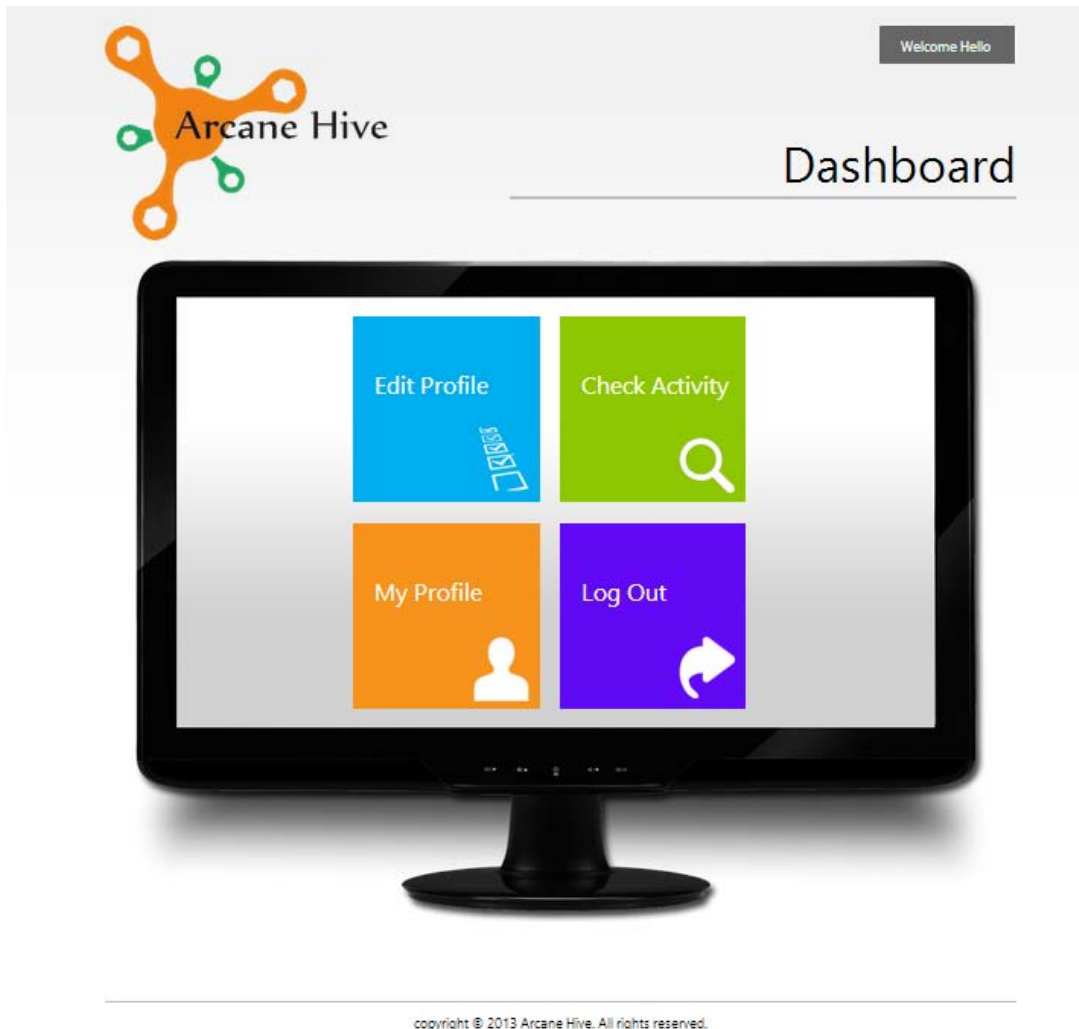



Figure 20: Homepage View of the website

From an employee's perspective, after logging in, he can see his basic informations and also has the facility to edit his informations. The employee can also check his activities in a graphical representation for a particular month. Moreover, the employee can check his activities of the previous months or even years till his joining date. Below the graphical representation, the employee is able to see a summary of his activities in a table for a particular month.

Dashboard Log Out

Arcane Hive

Employee Profile

Hello Nabil 

Profile Edit Profile Check Activity

Employee ID	1
Department	IT1
Designation	Junior Engineer
Mobile No	01674831893
Landphone No	8012394
Email	bapu6300@gmail.com
Joining Date	2013-06-02

copyright © 2013 Arcane Hive. All rights reserved.

Figure 21: Employee's Profile

On the other hand, from the employer's perspective, the employer can see his basic informations as well as his employees' informations. Moreover, the employer can also see the activities of his employees for any particular month or previous months in a graphical representation and also in a table for a summary view.

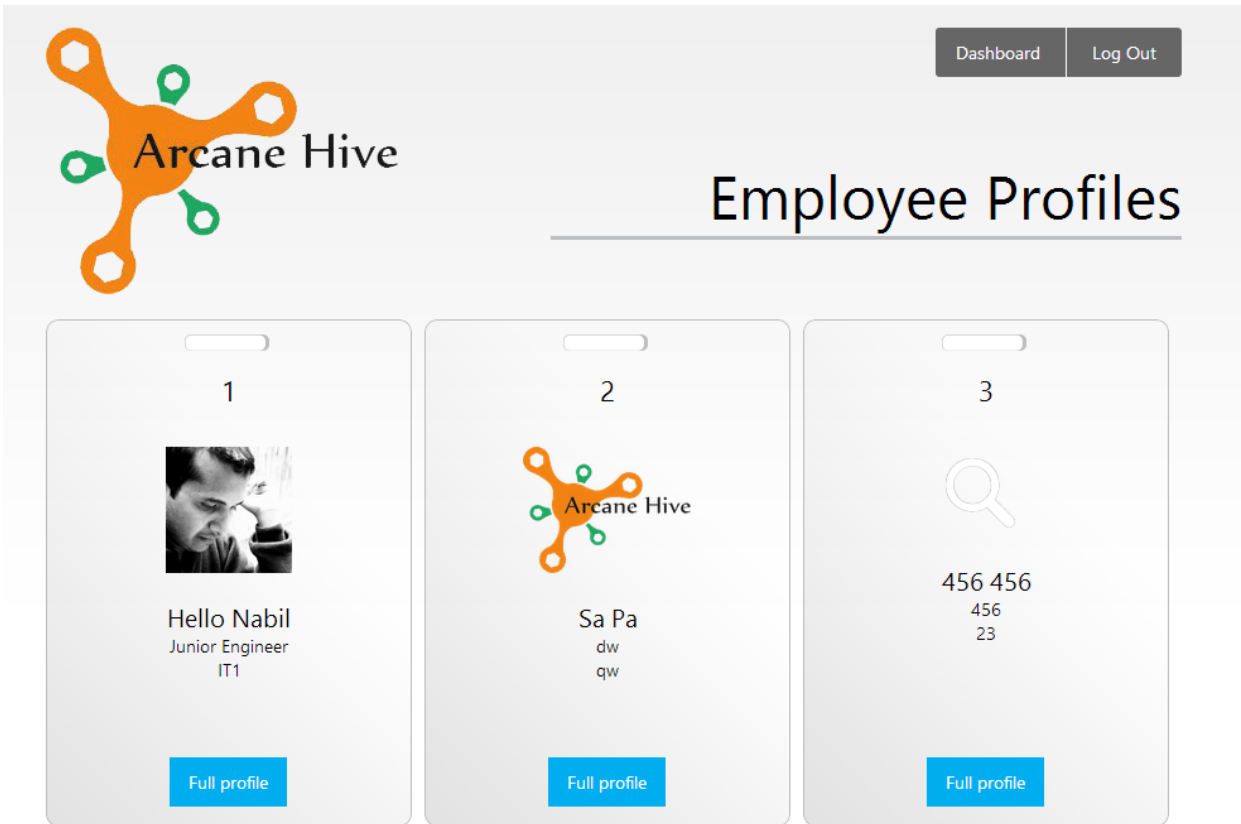


Figure 22: View of the employee list

3.Results and Discussion

3.1 Discussion on Desktop Application's Results

As we have already stated that Haar Classifier is used for face detection. As a result, our system shows a rectangle for each face it detects.

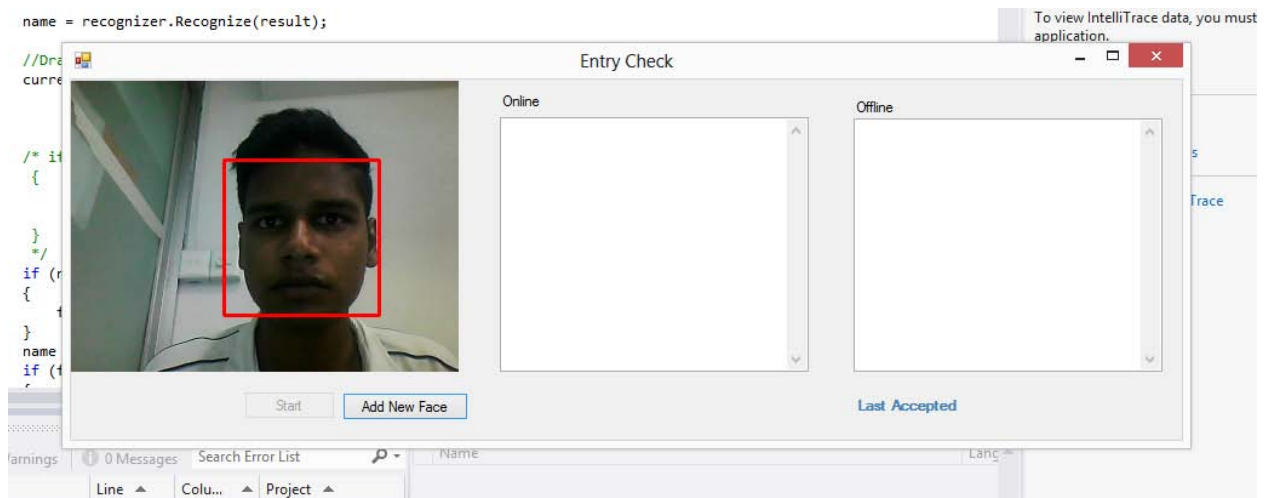


Figure 23: Face Detection

Moreover, PCA (Principal Component Analysis) is used for face detection in our system. After a face has been trained, our system draws a rectangle and the recognized user's name to indicate a face has been recognized. Our system can be made more efficient by making a database for trained faces so that the face is no longer need to be trained. After a face has been recognized, our proposed system will make the user online and his online activities will start to compute and will be recorded in the "time-table" database as stated before.

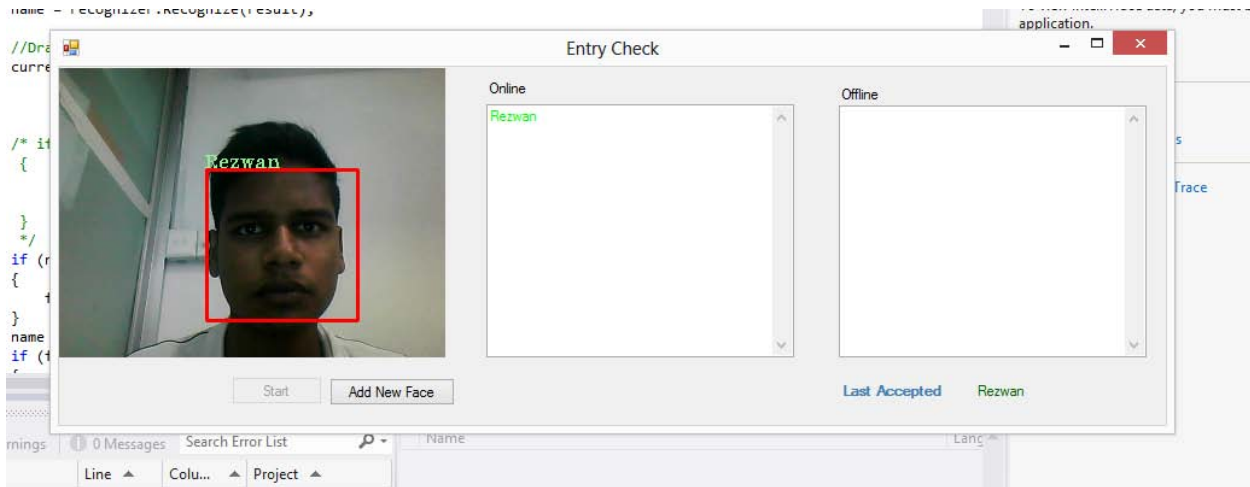


Figure 24: Face Recognition

When a face is not present in front of the camera and it results in no face being detected, and as the detection algorithm has been run for 15 times, the user being online will go offline immediately and the current time-stamp will be recorded as “End_entry” in the “time-table” database.

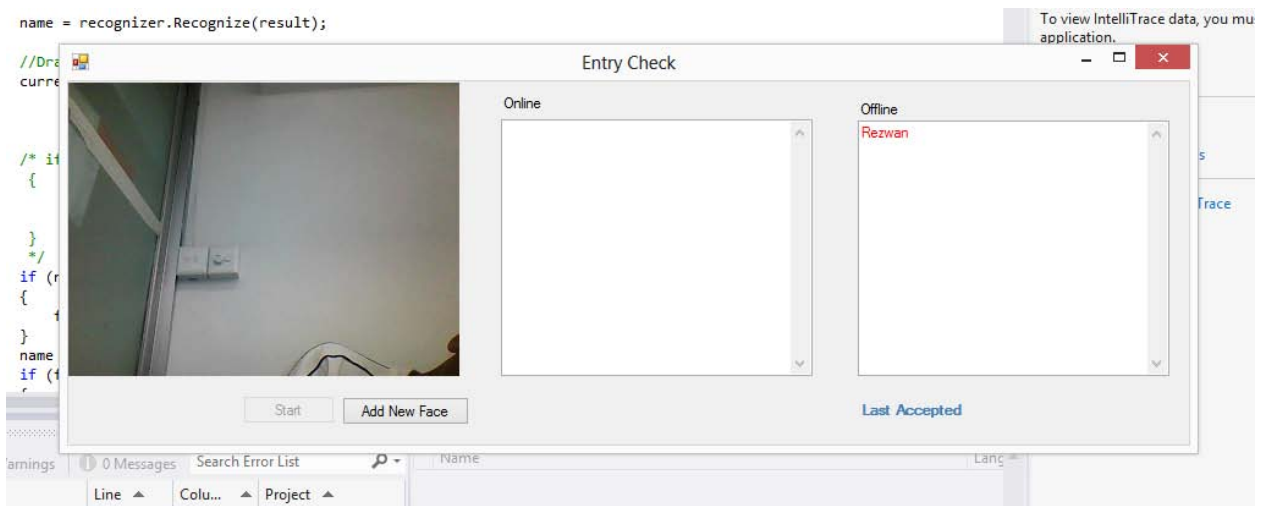


Figure 25: User going offline

3.2 Discussion on Web-based Application's Results

When a user has come online via face recognition and detection process, his records of entry time will start to be recorded in the "time-table" database. Similarly, when a user goes offline after not being detected via face detection algorithm, his records of end time will start to be recorded in the "time-table" database. As a result, we have a complete record of how long the employee has worked and how long he was away from his desk. Therefore, we represent this statistics in a graphical bar chart in our website. Moreover, we display the total summary of how long the user was working or away from his desk in a table for a particular month. The employer can also see the activities of the particular employee of the previous months or years.

In the image, the green bar shows how long the user was working on his desk and the red bar shows how long the user was away from his desk.

On Y-Axis, time is shown in one hour interval and on X-Axis, day of the month is shown in one day interval.

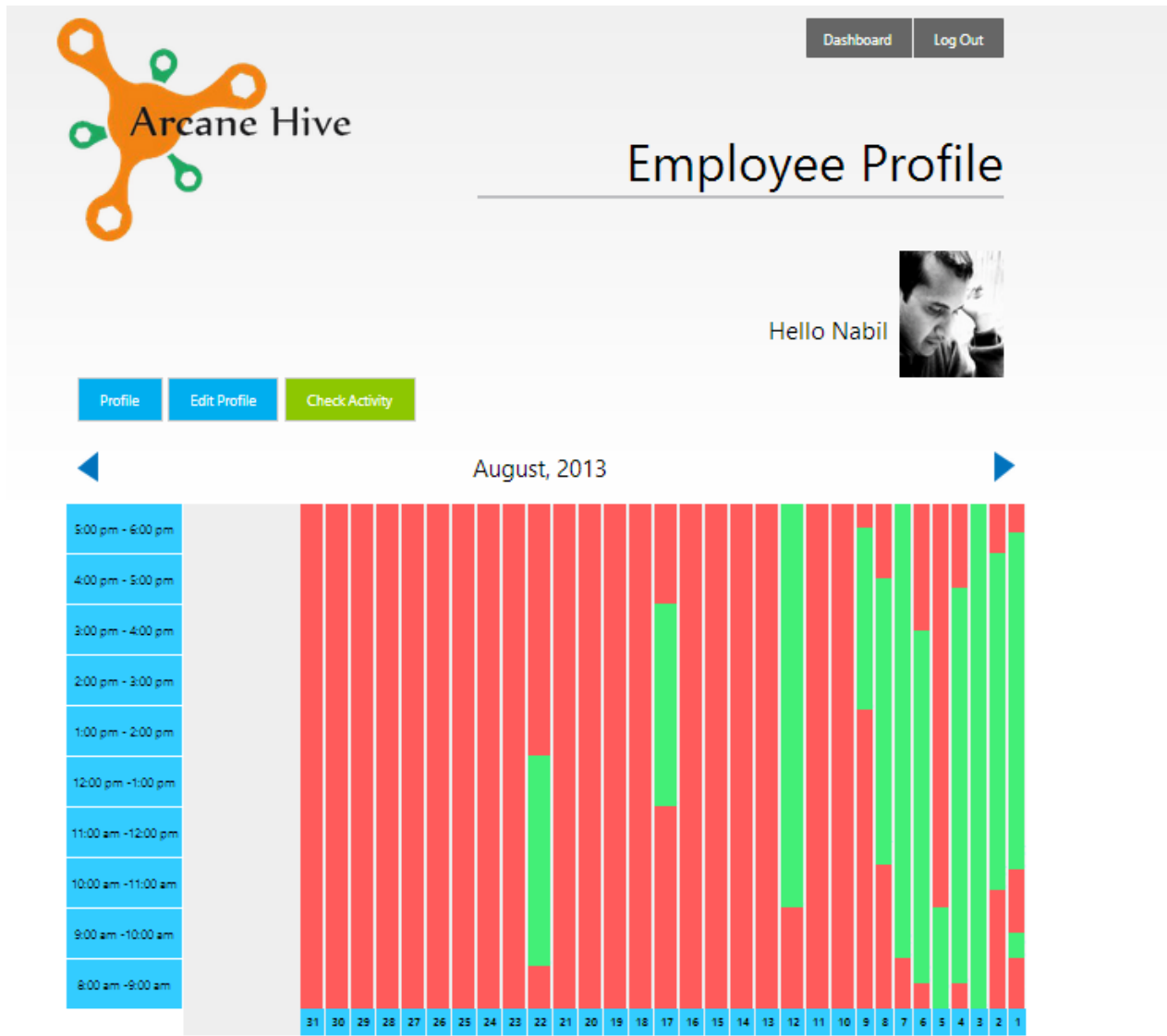


Figure 26: Graphical Representation of the employee activities

The total summary of how long the user was working and how long the user was away from his desk is shown in a table both in web-based application and android application like displayed below.

Activity Summary of August

Date	Online Activity Log	Offline Activity Log
1 August	6 Hr 45 Min	3 Hr 15 Min
2 August	6 Hr 40 Min	3 Hr 20 Min
3 August	10 Hr 0 Min	0 Hr 0 Min
4 August	7 Hr 50 Min	2 Hr 10 Min
5 August	2 Hr 0 Min	8 Hr 0 Min
6 August	6 Hr 58 Min	3 Hr 2 Min
7 August	9 Hr 0 Min	1 Hr 0 Min
8 August	5 Hr 40 Min	4 Hr 20 Min
9 August	3 Hr 35 Min	6 Hr 25 Min
10 August	0 Hr 0 Min	10 Hr 0 Min
11 August	0 Hr 0 Min	10 Hr 0 Min
12 August	8 Hr 0 Min	2 Hr 0 Min
13 August	0 Hr 0 Min	10 Hr 0 Min
14 August	0 Hr 0 Min	10 Hr 0 Min
15 August	0 Hr 0 Min	10 Hr 0 Min
16 August	0 Hr 0 Min	10 Hr 0 Min
17 August	4 Hr 0 Min	6 Hr 0 Min
18 August	0 Hr 0 Min	10 Hr 0 Min
19 August	0 Hr 0 Min	10 Hr 0 Min
20 August	0 Hr 0 Min	10 Hr 0 Min
21 August	0 Hr 0 Min	10 Hr 0 Min
22 August	4 Hr 10 Min	5 Hr 50 Min
23 August	0 Hr 0 Min	10 Hr 0 Min
24 August	0 Hr 0 Min	10 Hr 0 Min
25 August	0 Hr 0 Min	10 Hr 0 Min
26 August	0 Hr 0 Min	10 Hr 0 Min
27 August	0 Hr 0 Min	10 Hr 0 Min
28 August	0 Hr 0 Min	10 Hr 0 Min
29 August	0 Hr 0 Min	10 Hr 0 Min
30 August	0 Hr 0 Min	10 Hr 0 Min
31 August	0 Hr 0 Min	10 Hr 0 Min

copyright © 2013 Arcane Hive. All rights reserved.

Figure 27: Tabular Form of the employee activities



Figure 28: Tabular Form of the employee activities in Android Application

4. Conclusion and Future Work

In this chapter we represent the conclusion of this work. Further work is also analyzed suggesting various ways through which our present research may continue.

4.1 Conclusion

The thesis has described an efficient and affordable method for an employer to monitor his employees and observe their daily office activities. This system provides complete vision of an employer on his employees as he can check the daily office activities from anywhere he wants using the website and the android application. This system will bring ease to the employer to monitor his entire office-space and to take necessary actions which will enhance the efficiency level of an employee. In conclusion it is noted that our proposed system is able to detect multiple faces simultaneously and it can also recognize faces which already exist in the database by training the system. Thus, the entire system package clearly shows that it will assist to analyze the employee activity with ease.

Finally, experimental results are demonstrated to prove that this proposed system package is accurate and reliable to solve the promising and challenging real-time computer vision based employee activity analysis.

4.2 Future Works

We have already developed a desktop application, a web application and an android application. In near future there are a lot of scope where we can improve our current system package. Among all the future possibilities we are going to enhance our work on the following:

- a) Firstly we will look forward to the data representation in our website. It will be more user-friendly.
- b) Our android application will definitely contain the graphical view of time spent in front of desk and time spent away from the desk
- c) We will increase the efficiency of our PCA Algorithm by training the system. Faces which have already been detected will be stored in the database for future recognition. As our database gets rich the efficiency will increase automatically.
- d) Implement the use of threads which will allow us to do parallel processing, pipelining. This will increase our hardware performance which will obviously increase the total system performance.

References

1. J. G..RoshanTharanga, S. M. S. C. Samarakoon, T. A. P. Karunarathne, K. L. P. M. Liyanage, M. P. A. W. Gamage, D. Perera, "SMART ATTENDANCE USING REAL TIME FACE RECOGNITION (SMART - FR)", Department of Electronic and Computer Engineering, Sri Lanka Institute of Information Technology (SLIIT), Malabe, Sri Lanka.
2. H. K. Ekenel, J. Stallkamp, H. Gao, M. Fischer, R. Stiefelhagen, "FACE RECOGNITION FOR SMART INTERACATINONS", interACT Research, Computer Science Department, Universit"atKarlsruhe (TH).
3. Kyungham Kim, "Face Recognition using Principle Component Analysis", Department of Computer Science, University of Maryland, College Park, MD 20742, USA
4. Phillip Ian Wilson, and Dr. John Fernandez, "FACIAL FEATURE DETECTION USING HAAR CLASSIFIERS", Texas A&M University.
5. The Code Project, "EMGU Multiple Face Recognition using PCA and Parallel Optimisation", 05 October 2011.
6. M. Gopi Krishna, A. Srinivasulu, Prof (Dr.) T.K.Basak, "Face Detection System on Ada boost Algorithm Using Haar Classifiers", International Journal of Modern Engineering Research (IJMER), Vol.2, Issue.6, Nov-Dec. 2012 pp-3996-4000, ISSN:2249-6645, Department of Electronics (E.C.E), Vaagdevi Institute of Tech & Science, Peddasettipalli(Village), Proddatur(M.D), Kadapa(Dist), Andhra Pradesh A.P, India. Dean (R&D), JIS College of Engineering, Kalyani,W.B.
7. Kyungham Kim, "Face Recognition using Principle Component Analysis", Department of Computer Science University of Maryland, College Park MD 20742, USA

8. Phillip Ian Wilson, Dr. John Fernandez, FACIAL FEATURE DETECTION USING HAAR CLASSIFIERS, Texas A&M University – Corpus Christi
9. Adolf, F. How-to build a cascade of boosted classifiers based on Haar-like features.
10. Bradski, G. Computer vision face tracking for use in a perceptual user interface. Intel Technology Journal, 2nd Quarter, 1998.
11. <http://www.mybringback.com/tutorial-series/12924/android-tutorial-using-remote-databases-php-and-mysql-part-1/>
12. <http://developer.android.com>
13. <http://www.androiddesignpatterns.com/2012/06/app-force-close-honeycomb-ics.html>
14. <http://developer.android.com/reference/android/widget/Button.html>
15. <http://suvendugiri.wordpress.com/2012/01/31/android-using-button-click-event-with-example/>
16. <http://stackoverflow.com/questions/10146028/dynamic-button-onclick-event-inside-for-loop>
17. <http://stackoverflow.com/questions/6612229/why-i-can-not-set-onclicklistener-for-a-button-in-a-dialog-view>