CHAPTER- 5

IMPLEMENTATION

5.1 Class diagram

The class diagram is a static diagram. It represents the static view of an application. Class diagram not only used for visualizing, describing and documenting different aspects of system but also for constructing executable code of the software application. The class diagram describes the attributes and operation of a class and the constraints imposed on the system. Fig 5.3 shows the class diagram.

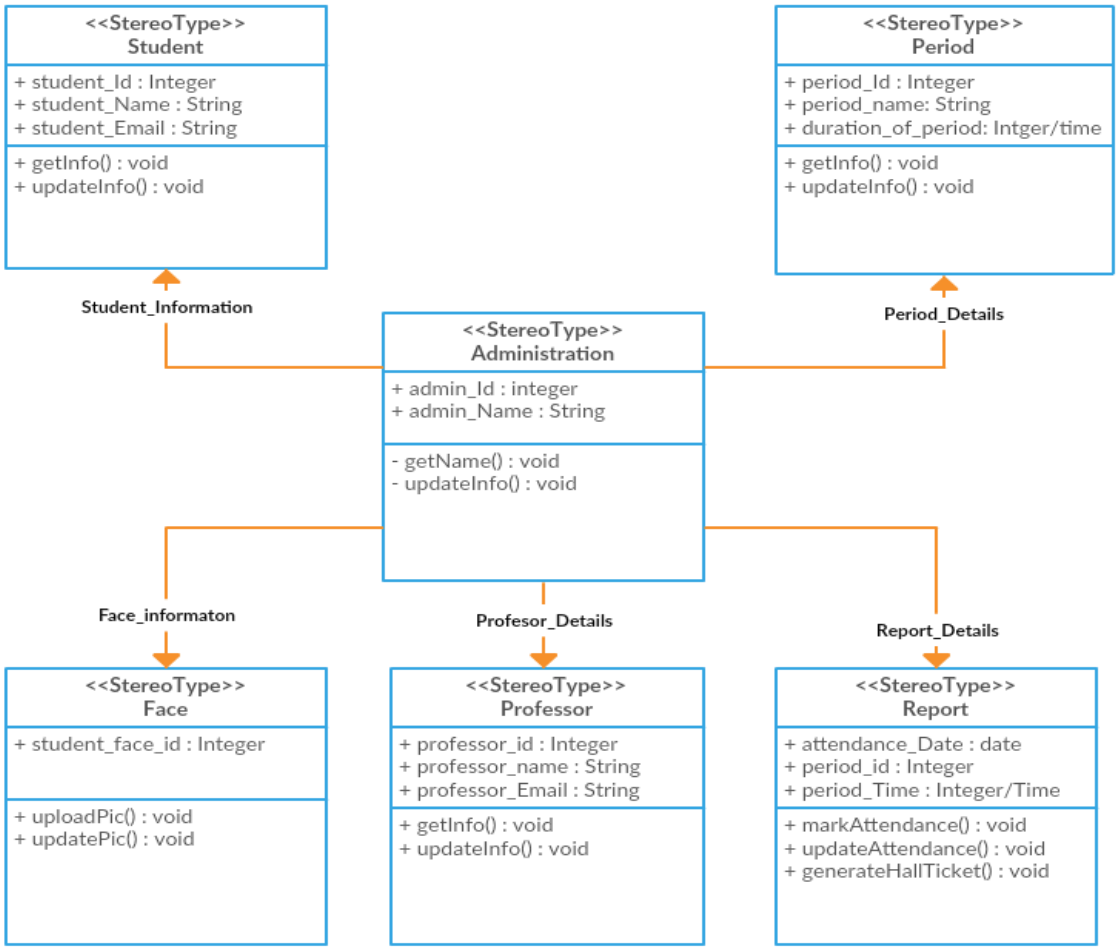


Figure 5.1 Class Diagram

5.1.1 List of modules

This project contains four modules, those are below.

- **Module 1:** (Image processing)

  - Image processing.
  - Use Machine learning algorithm for image processing.

5.1.1.2 Module 2: (To connect Database)

- Store the captured student information to database.
- Process the data for required processing.
- Providing administration privileges using existing data
- Adding or removing professor details.

5.1.1.3 Module 3: (IoT)

- Camera to remote system using internet
- Remote system will process those data

5.3.1.4 Module 4: (Hall ticket generation)

- Hall ticket generation.

5.1.1.1Module 1: (Image Processing)

Image of student will be processed using Machine learning algorithm. This project uses LBPH algorithm for train and recognition of Student images. Face detection and recognition will be done using Face class. It is shown in Fig:5.1.3

There are different types of face recognition algorithms, for example:

- Eigenfaces (1991)
- Local Binary Patterns Histograms (LBPH) (1996)
- Fisherfaces (1997)
- Scale Invariant Feature Transform (SIFT) (1999)
- Speed Up Robust Features (SURF) (2006)

In this project uses the LBPH algorithm to recognize the image,

Local Binary Pattern

Local binary patterns (LBP) is a type of visual descriptor used for classification in computer vision. LBP is the particular case of the Texture Spectrum model proposed in 1990. LBP was first described in 1994. It has since been found to be a powerful feature for texture classification; it has further been determined that when LBP is combined with the Histogram of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets.

The LBP feature vector

Divide the examined window into cells (e.g. 16x16 pixels for each cell).For each pixel in a cell, compare the pixel to each of its 8 neighbors (on its left-top, left-middle, left-bottom, right-top, etc.). Follow the pixels along a circle, i.e. clockwise or counter-clockwise. Where the center pixilates value is greater than the neighbor value, write"0". Otherwise, write "1". This gives an 8-digit binary number (which is usually converted to decimal for convenience).Compute the histogram, over the cell, of the frequency of each "number" occurring (i.e., each combination of which pixels are smaller and

which are greater than the center). This histogram can be seen as a 256-dimensional feature vector. Optionally normalize the histogram. Concatenate (normalized) histograms of all cells. This gives a feature vector for the entire window.

Correlation

Correlation technique is used for face recognition. Where after face detection image under- goes face recognition process, where test image will be compared with training images in order to perform face recognition.

Parameters

- **Radius:** the radius is used to build the circular local binary pattern and represents the radius around the central pixel. It is usually set to 1.
- **Neighbors:** the number of sample points to build the circular local binary pattern.
- **Grid X:** the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.
- **Grid Y:** the number of cells in the vertical direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

Training the Algorithm

First, system need to train the algorithm. To do so, system need to use a dataset with the facial images of the people system want to recognize. system need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and system will give an output. Images of the same person must have the same ID. With the training set already constructed, the LBPH computational steps follows.

Applying the LBP operation

The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameters radius and neighbors.

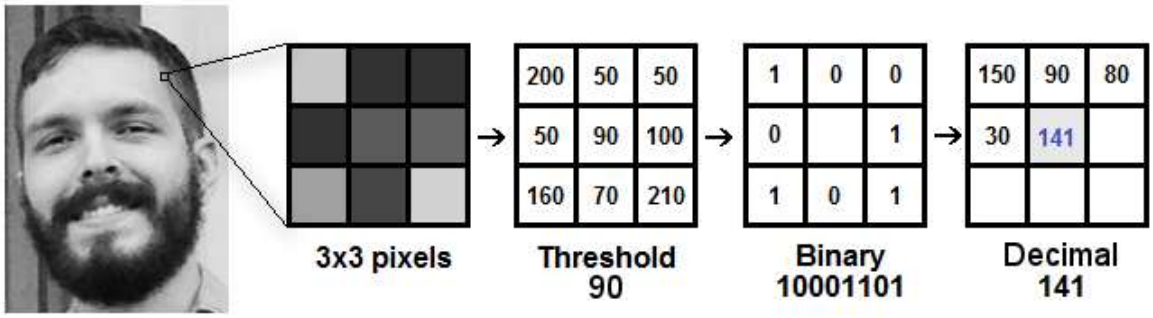


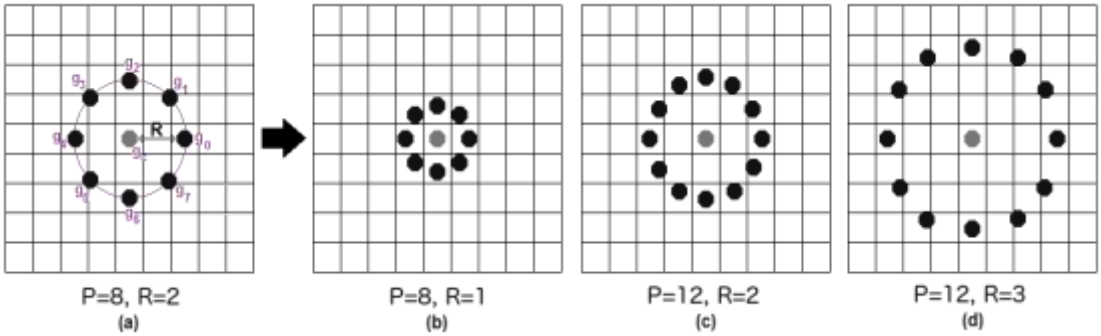
Figure 5.1.1 Pixel matching

Based on the image above, break it into small steps,

- Suppose system have a facial image in grayscale.
- System can get part of this image as a window of 3x3 pixels.
- It can also be represented as a 3x3 matrix containing the intensity of each pixel (0~255).
- Then, system need to take the central value of the matrix to be used as the threshold.
- This value will be used to define the new values from the 8 neighbors.
- For each neighbor of the central value (threshold), system set a new binary value. System set 1 for values equal or higher than the threshold and 0 for values lower than the threshold.
- Now, the matrix will contain only binary values (ignoring the central value). System need to concatenate each binary value from each position from the matrix line by line into a new binary value (e.g. 10001101). Note: some authors use other approaches to concatenate the binary values (e.g. clockwise direction), but the final result will be the same.
- Then, system convert this binary value to a decimal value and set it to the central value of the matrix, which is actually a pixel from the original

image.

- At the end of this procedure (LBP procedure), system have a new image which represents better the characteristics of the original image.
- Note: The LBP procedure was expanded to use a different number of radius and neighbors, it is called Circular LBP.



It can be done by using If some data point is between the pixels, it uses the values from the 4 nearest pixels (2x2) to estimate the value of the new data point.

Extracting the Histograms

Now, using the image generated in the last step, system can use the Grid X and Grid Y parameters to divide the image into multiple grids, as can be seen in the following image:

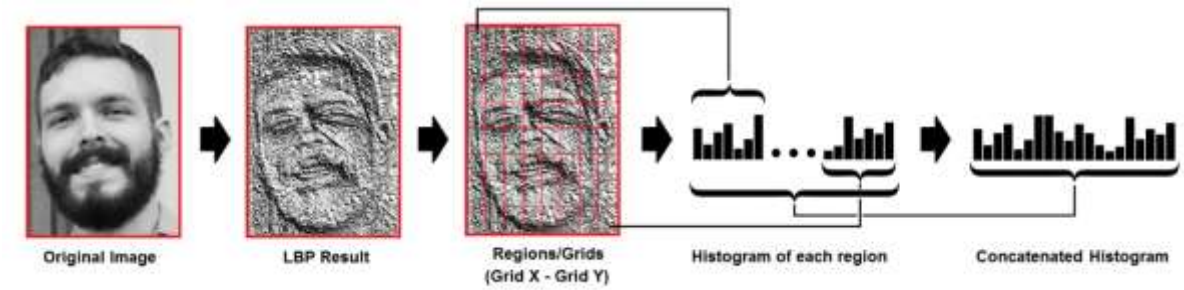


Figure 5.1.2 Pixel matching with histogram

Based on the image above, system can extract the histogram of each region as follows:

As we have an image in grayscale, each histogram (from each grid) will contain only 256 positions (0~255) representing the occurrences of each pixel intensity.

Then, we need to concatenate each histogram to create a new and bigger histogram. Supposing we have 8x8 grids, we will have 8x8x256=16.384 positions in the final histogram. The final histogram represents the characteristics of the image original image.

Performing the face recognition

In this step, the algorithm is already trained. Each histogram created is used to represent each image from the training dataset. So, given an input image, system will perform the steps again for this new image and creates a histogram which represents the image.

So to find the image that matches the input image system just need to compare two histograms and return the image with the closest histogram.

System can use various approaches to compare the histograms (calculate the distance between two histograms), for example: euclidean distance, chi-square, absolute value, etc. In this example, system can use the Euclidean distance (which is quite known) based on the following formula:

$$D = \sqrt{\sum_{i=1}^{n}(hist1_i - hist2_i)^2}$$

So the algorithm output is the ID from the image with the closest histogram. The algorithm should also return the calculated distance, which can be used as a ‘confidence’ measurement. Note: don’t be fooled about the ‘confidence’ name, as lower confidences are better because it means the distance between the two histograms is closer.

System can then use a threshold and the ‘confidence’ to automatically estimate if the algorithm has correctly recognized the image. system can assume that the algorithm has successfully recognized if the confidence is lower than

the threshold defined.

Need to use LBPH algorithm

- LBPH is one of the easiest face recognition algorithms.
- It can represent local features in the images.
- It is possible to get great results (mainly in a controlled environment).
- It is robust against monotonic gray scale transformations.
- It is provided by the OpenCV library (Open Source Computer Vision Library).

a) Face class

Face class will get the picture of student and process those student picture to recognice student picture.

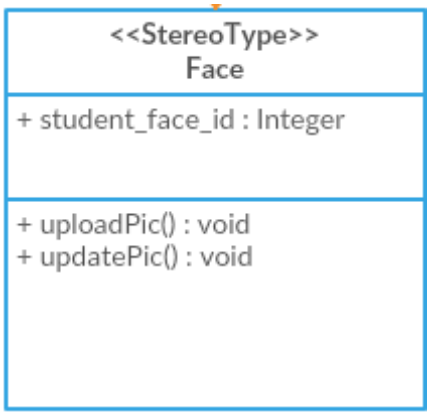


Figure 5.1.3 Face Class Diagram

```
def uploadPic():
    #updating Student picture
def updatePic():
    #updating Student picture
def train():
    #Training of student
```

Haar Cascade

Haar Cascade is a machine learning object detection algorithm used to identify objects in an image or video and based on the concept of features proposed by Paul Viola and Michael Jones in their paper "Rapid Object

Detection using a Boosted Cascade of Simple Features" in 2001.

It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

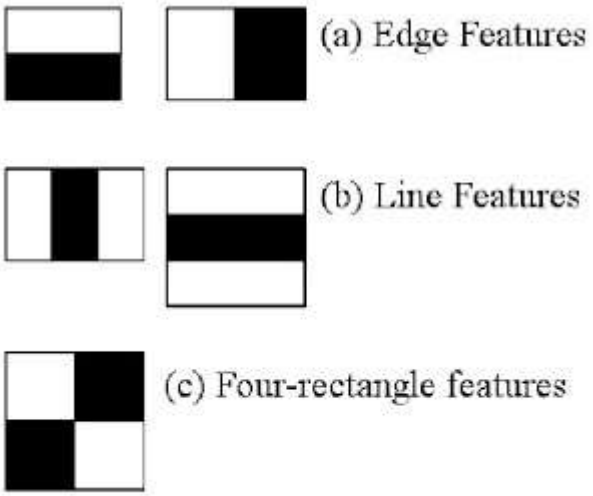
The algorithm has four stages:

- Haar Feature Selection
- Creating Integral Images
- Adaboost Training
- Cascading Classifiers

It is well known for being able to detect faces and body parts in an image, but can be trained to identify almost any object.

Lets take face detection as an example. Initially, the algorithm needs a lot of positive images of faces and negative images without faces to train the classifier. Then we need to extract features from it.

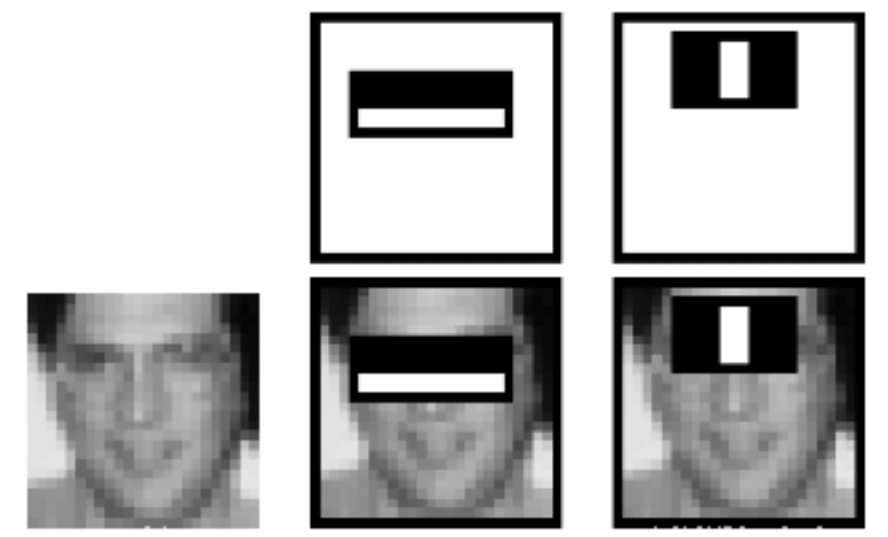
First step is to collect the Haar Features. A Haar feature considers adjacent rectangular regions at a specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums.



Integral Images are used to make this super fast.

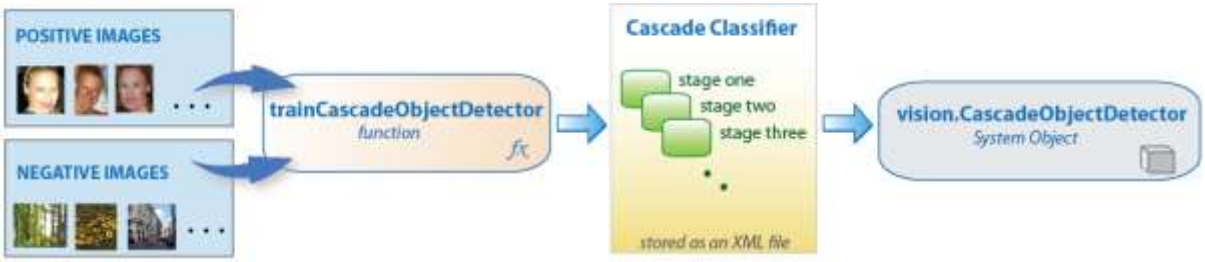
But among all these features we calculated, most of them are irrelevant. For example, consider the image below. Top row shows two good features. The first feature selected seems to focus on the property that the region of the eyes is often darker than the region of the nose and cheeks. The second feature selected relies on the property that the eyes are darker than the bridge of the nose.

But the same windows applying on cheeks or any other place is irrelevant.



During the detection phase, a window of the target size is moved over the input image, and for each subsection of the image and Haar features are calculated. This difference is then compared to a learned threshold that separates non-objects from objects. Because each Haar feature is only a "weak classifier" (its detection quality is slightly better than random guessing) a large number of Haar features are necessary to describe an object with sufficient accuracy and are therefore organized into cascade classifiers to form a strong classifier.

Cascade Classifier



The cascade classifier consists of a collection of stages, where each stage is an ensemble of weak learners. The weak learners are simple classifiers called decision stumps. Each stage is trained using a technique called boosting. Boosting provides the ability to train a highly accurate classifier by taking a weighted average of the decisions made by the weak learners.

Each stage of the classifier labels the region defined by the current location of the sliding window as either positive or negative. Positive indicates that an object was found and negative indicates no objects were found. If the label is negative, the classification of this region is complete, and the detector slides the window to the next location. If the label is positive, the classifier passes the region to the next stage. The detector reports an object found at the current window location when the final stage classifies the region as positive.

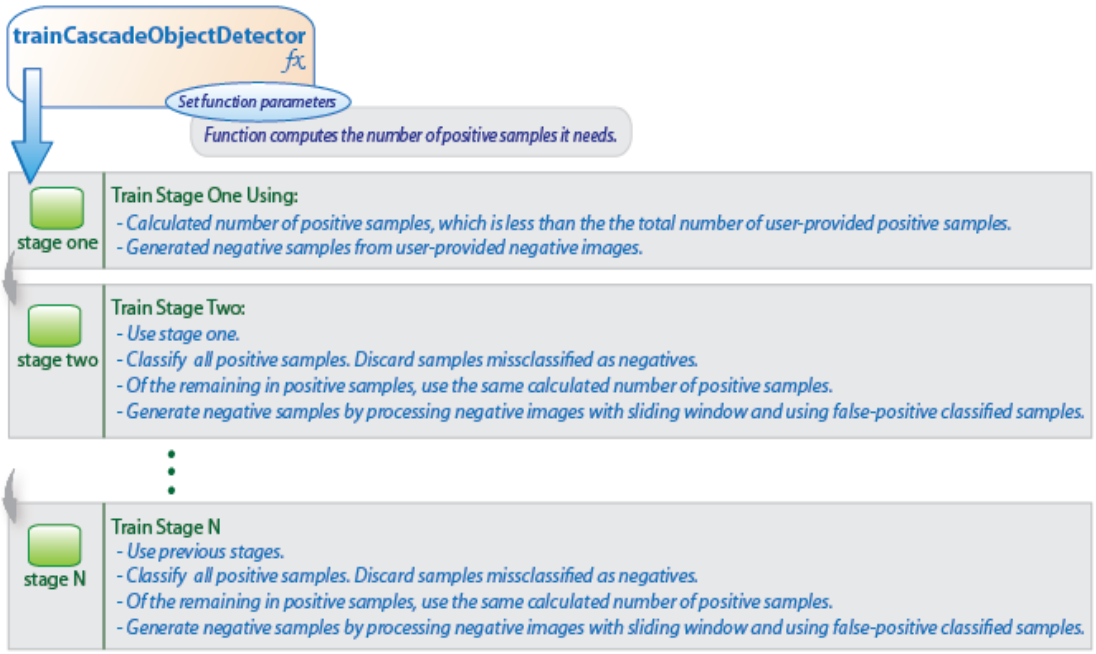
The stages are designed to reject negative samples as fast as possible. The assumption is that the vast majority of windows do not contain the object of interest. Conversely, true positives are rare and worth taking the time to verify.

- A true positive occurs when a positive sample is correctly classified.
- A false positive occurs when a negative sample is mistakenly classified as positive.
- A false negative occurs when a positive sample is mistakenly classified as negative.

To work well, each stage in the cascade must have a low false negative rate. If a stage incorrectly labels an object as negative, the classification stops, and then cannot correct the mistake. However, each stage can have a high false positive rate. Even if the detector incorrectly labels a non-object as positive, classifier correct the mistake in subsequent stages. Adding more stages reduces the overall false positive rate, but it also reduces the overall true positive rate.

Cascade classifier training requires a set of positive samples and a set of negative images. Provide a set of positive images with regions of interest

specified to be used as positive samples. Use the Image Labeler to label objects of interest with bounding boxes. The Image Labeler outputs a table to use for positive samples. Provide a set of negative images from which the function generates negative samples automatically. To achieve acceptable detector accuracy, set the number of stages, feature type, and other function parameters.



5.1.1.2 Module 2: (Database)

Information of student, professor, administration details, Period details will be stored into database such as SQlite database and CSV file. This phase will be done using Administration class, Student class, Professor class, and Period class.

Administration class

Administration class will authenticate the user of application.

Figure 5.1.4 shows the administration class diagram

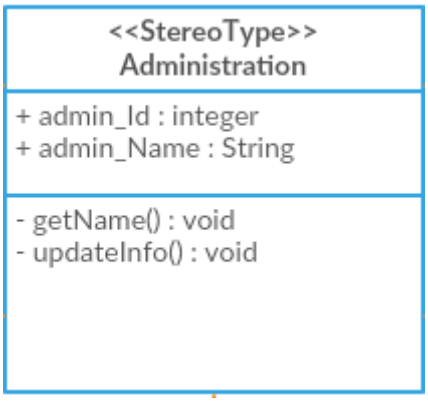


Figure 5.1.4 Administration Class Diagram

```
def getName:
        admin_Id
        admin_Name:
        #Gathering Information of Admin
def updateInfo():
        admin_Id:
        admin_Name:
        #Updating Admin information
```

b) Student class

Student class will get information about the student and also update information of student. Figure 5.1.5 shows the student class diagram.

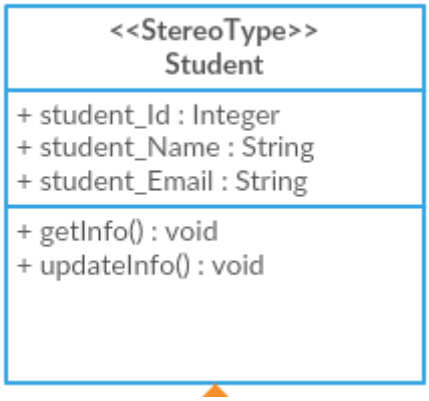


Figure 5.1.5 Student Class Diagram

```
def getName():
        admin_Id
        admin_Name:
        #Gathering Information of Student
def updateInfo():
        admin_Id:
```

admin_Name:
#Updating Student information by Student class

c) Professor class

Period information will be gathered using Professor class
Figure 5.1.6 shows the professor class diagram

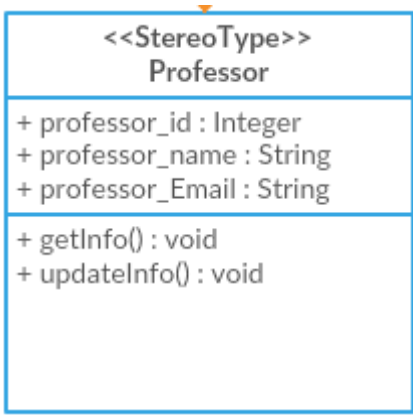


Figure 5.1.6 Professor class diagram

def getName:
admin_Id
admin_Name:
#Gathering Information of Professor
def updateInfo():
admin_Id:
admin_Name:
#Updating information of Professor

d) Period class

Period details and Professor details who taking the class in the class romm that are updated by the Period class. Figure 5.1.7 shows the period class diagram

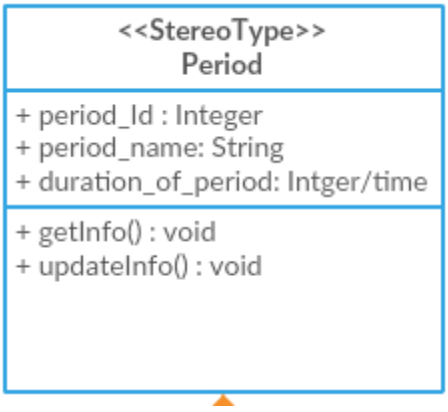


Figure 5.1.7 Period class diagram

def getInfo():

#getting period information

def updateInfo():

#updating period information

5.1.1.3 Module 3: (IoT)

It's actual process of detecting the image with use of IP camera and internet applications.

5.1.1.4 Module 4: (Hall-Ticket generation)

Hall ticket of student will be generated if attendance percentage is above 80. It will done by the Report class. It is shown in Fig:5.3.8

e) Report class

Report of student information generated and it will sent the details to related Professors.

It's also generate the hall ticket student who got the attendance percentage above 80%.

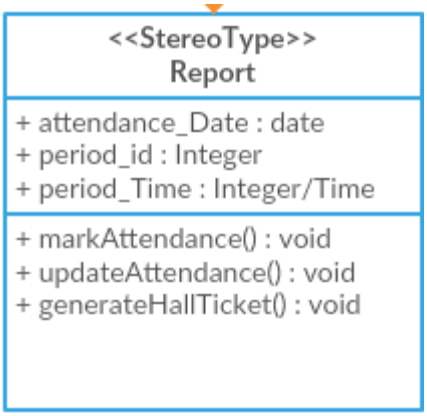


Figure 5.1.8 Report class diagram

def markAttendance():

#marking student attendance by image processing

def updateAttendance();

#updating student information in case of problem on remote system.

Def generateHallTicket():

#it will generate the hall ticket if attendance percentage 80% above.