

# **FACE RECOGNITION BASED STUDENT ATTENDANCE SYSTEM**

## **A MAJOR PROJECT REPORT**

**Submitted In the partial fulfilment  
of the requirement for the award of  
degree of**

**Bachelor of Vocation**

**In**

**Software Development**

**Submitted by  
Mahesh Bisht**

**00724818117**

**B.voc(SD) 6<sup>th</sup> Semester**



**AMBEDKAR INSTITUTE OF TECHNOLOGY  
GOVT. OF NCT DELHI**

**Affiliated to  
Guru Gobind Singh Indraprastha University  
Dwarka, Sector 16, New Delhi  
2019-2020**

## **ACKNOWLEDGEMENT**

I found this golden chance to acknowledge all those people who had blessed, encouraged and supported me technically and morally through all the phases of my project. I take this opportunity to express my profound sense of gratitude. I thank all mighty GOD for giving me this opportunity to express gratitude to all those who helped me in successful completion of this project.

I pay my immense gratitude to for providing help and giving me a chance for showing my skills through continued support and co-operation during the concerned project. I am deeply indebted to staff of Computer Dept., for their sincere co-operation and sparing time to answer questionnaires with their selfless efforts and co-operation because of which I am able to complete this project.

I want to thank all from the core of my heart to provide the entire infrastructure throughout the course.

I am deeply indebted to my parents who have always been a perennial source of information, encouragement and inspiration for entire education required.

## **ABSTRACT**

**“Face Recognition Based Student Attendance System”** is a system that is used to mark the attendance of students. The Traditional Attendance System had many limitations such as marking attendance in the register, papers, or sheets and time-consuming work. This Attendance System will overcome these limitations.

This report will describe the analysis, design, and implementation of the **"Face Recognition Based Student Attendance System"** that uses student face to mark attendance.

The **“Face Recognition Based Student Attendance System”** deals with the maintenance of the student’s attendance details. It will be able to mark attendance based on available classes. It is maintained on the daily basis of their attendance. The teacher will be provided with a separate username & password to make the student’s status.

The teacher handling the particular subjects responsible to make the class time for the subject. Only if the student is present in that particular period, the attendance will be marked. The student's attendance reports-based attendance will be generated.



## **AMBEDKAR INSTITUTE OF TECHNOLOGY**

### **Student Declaration**

This is to certify that the project entitled “**Face Recognition Based Student Attendance System**” completed during the session 2019-2020 for B.voc(SD) degree is in good faith of research work and all the sources used to complete this project are duly acknowledged. In case the project report, or any part of it, is found to be copied or quoted without reference, I shall be solely held accountable for the repercussions arising there from.

**Dated: .....**

**Signature of the Student**



## **AMBEDKAR INSTITUTE OF TECHNOLOGY**

### **Supervisor Certificate**

The project entitled “**Face Recognition Based Student Attendance System**” by Mahesh Bisht has been completed under my supervision.

**Dated:** .....

**Signature of the Supervisor**

## TABLE OF CONTENTS

Chapter No	Name	PAGE NO.
1	INTRODUCTION	1
1.1	INTRODUCTION	1
1.2	DOCUMENT PURPOSE	1
1.3	PROBLEM STATEMENT	2
1.4	PROJECT SCOPE	2
1.5	AIMS AND OBJECTIVES	2
1.6	PROJECT CATEGORY	2
2	SYSTEM ANALYSIS	3
2.1	INTRODUCTION	3
2.2	EXISTING SYSTEM	3
2.3	PROPOSED SYSTEM:	3
2.4	FEASIBILITY STUDY	4
	2.4.1 ECONOMICALLY FEASIBILITY	4
	2.4.2 TECHNICAL FEASIBILITY	4
	2.4.3 OPERATIONAL FEASIBILITY	4
3	SYSTEM SPECIFICATION	5
3.1	HARDWARE REQUIREMENTS (Minimum Requirement)	5
3.2	SOFTWARE REQUIREMENTS (Minimum Requirement)	5
4	PROJECT DESCRIPTION	7
4.1	PROBLEM DEFINITION:	7
4.2	PROJECT OVERVIEW:	
4.3	MODULE DESCRIPTION	7
	4.3.1 ADMINISTRATOR MODULE	7
	4.3.2 TEACHER MODULE	8
	4.3.3 STUDENT MODULE	8
4.4	SYSTEM FLOW DIAGRAM	9
4.5	DATA FLOW DIAGRAM	10
	4.5.1 DFD LEVEL 0:	10
	4.5.2 DFD LEVEL 1:	11
	4.5.2.1 ADMIN DFD 1	11
	4.5.2.2 TEACHER DFD 1	12
	4.5.2.3 STUDENT DFD 1	13
4.6	SYSTEM DESIGN:	14
	4.6.1 ENTITY RELATIONSHIP DIAGRAM	14
	4.6.2 USE CASE DIAGRAM	15
	4.6.2 DATABASE DESIGN	16
	4.6.2.1 USER TABLE	16
	4.6.2.2 ATTENDANCE TABLE	16
	4.6.2.3 COURSE TABLE	16
	4.6.2.4 SUBJECT TABLE	17

	4.6.2.5	CREATE CLASS TIMETABLE	17
	4.6.3	INPUT DESIGN	18
	4.6.4	OUTPUT DESIGN	18
5		SYSTEM TESTING	19
	5.1	INTRODUCTION	19
	5.2	TESTING METHODOLOGIES:	19
	5.2.1	UNIT TESTING:	19
	5.2.2	INTEGRATION TESTING	20
	5.2.3	SYSTEM TESTING:	20
	5.2.4	DEFECT TESTING	20
	5.2.5	BLACK-BOX TESTING	20
	5.3	TEST CASES	21
6		SYSTEM IMPLEMENTATION	22
	6.1	PURPOSE & IMPLEMENTATION	22
	6.2	SYSTEM MAINTENANCE	23
7		CONCLUSION AND FUTURE ENHANCEMENT	24
8		APPENDICES	25
	8.1	SOURCE CODE	25
	8.2	SCREEN SHOT	38
9		BIBLIOGRAPHY	47

# CHAPTER 1

## INTRODUCTION

### 1.1 INTRODUCTION

**“Face Recognition based Student Attendance System”** is used by college student to mark attendance. Teachers are also able to mark student attendance. Attendance System is a system that collects student attendance details. An Attendance System is a bridge between students and teachers. An Attendance System refers to aspects of student attendance: mark attendance, check attendance, check the class time, student profile check, teacher profile check, and create class time.

**“Face Recognition based Student Attendance System”** is a web application developed for maintaining the attendance of the student on the daily basis in the collage. Here the teachers, who are handling the subjects, will have to register themselves. Each teacher will have to give a separate username and password. Teachers are responsible for making class time so that students can mark their attendance. Students also have to register themselves with their enrollment numbers. Students have to mark their attendance based on class time.

### 1.2 DOCUMENT PURPOSE

The purpose of this Attendance System is to provide a system that can be used to mark Student attendance and analyze it.

Attendance System does not make decisions or manage operations, they provide the information to the Admin and the teachers who make more accurate and timely decisions to manage their operations.

Attendance shows the student regularity but many colleges do not satisfy with the traditional attendance system of marking attendance. That's why the modern system should be implemented to mark attendance that will provide results more quickly and efficiently.



### **1.3 PROBLEM STATEMENT**

The basic problems that teachers face are:

- Manual entry for the students is a very tedious job to maintain the record for the students.
- The human effort is more here.
- The attendance will be carried out in the handwritten registers.
- The retrieval of information is not easy.

### **1.4 PROJECT SCOPE**

This project traverses a few areas of the attendance system and is required to perform several kinds of research to be able to achieve the project objectives. The area covers include:

- Attendance marking system using Face Detection Approach.
- Python Technology is used for the development of the application.
- A web-based platform means that the system will be available for access 24/7 except when there is a temporary server issue.

### **1.5 AIMS AND OBJECTIVES**

Specific goals are:

- To produce a web-based platform that allows students and teachers to register and mark attendance of students.
- To ease the tedious process of attendance system.
- To provide an easy to use interface.
- Students have to use their faces to mark attendance.

### **1.6 PROJECT CATEGORY**

This “Face Recognition Based Student Attendance System” is a web-based application. The nature of this software is to handle a particular task that’s why it comes under the category of application software. This project falls under the category of internet technologies with RDBMS, Relations database management system. This project is utilizing a relational database as back End. We are using the Django Framework to develop this software.

Django is a python-based web development framework. That is especially suited for web development and can be embedded into HTML. Python has rapidly increased its influence and can be used almost anywhere.

## **CHAPTER 2**

### **SYSTEM ANALYSIS**

#### **2.1 INTRODUCTION**

Analysis can be defined as breaking up of any whole so as to find out their nature, function etc. It defines design as to make preliminary sketches of; to sketch a pattern or outline for plan. To plan and carry out especially by artistic arrangement or in a skillful way. System analysis and design can be characterized as a set of techniques and processes, a community of interests, a culture and an intellectual orientation. The various tasks in the system analysis include the following.

- Understanding application.
- Planning.
- Scheduling.
- Developing candidate solution.
- Performing trade studies.
- Performing cost benefit analysis.
- Recommending alternative solutions.
- Selling of the system.
- Supervising, installing and maintaining the system.

#### **2.2 EXISTING SYSTEM**

The Existing system is a manual entry for the students. Here the attendance will be carried out in the handwritten registers. It will be a tedious job to maintain the record for the user. The human effort is more here. The retrieval of the information is not as easy as the records are maintained in the handwritten registers.

This application requires correct feed on input into the respective field. Suppose the wrong inputs are entered, the application resists working. so, the user finds it difficult to use.

#### **2.3 PROPOSED SYSTEM:**

To overcome the drawbacks of the existing system, the proposed system has been evolved. This project aims to reduce the paper work and saving time to mark attendance by the students and teachers. The system provides with the best user interface. The accurate attendance can be marked by using this proposed system.

### **2.3.1 Advantages of Proposed System**

- It is trouble-free to use.
- It is a relatively fast approach to enter attendance.
- Is highly reliable.
- Best user Interface.

## **2.4. FEASIBILITY STUDY:**

Feasibility analysis begins once the goals are defined. It starts by generating broad possible solutions, which are possible to give an indication of what the new system should look like. This is where creativity and imagination are used. Analysts must think up new ways of doing things- generate new ideas. There is no need to go into the detailed system operation yet. The solution should provide enough information to make reasonable estimates about project cost and give users an indication of how the new system will fit into the organization. It is important not to exert considerable effort at this stage only to find out that the project is not worthwhile or that there is a need significantly change the original goal.

Feasibility of a new system means ensuring that the new system, which we are going to implement, is efficient and affordable. There are various types of feasibility to be determined. They are

### **2.4.1 ECONOMICALLY FEASIBILITY**

Development of this application is highly economically feasible. The only thing to be done is making an environment with an effective supervision. It is cost effective in the sense that has eliminated the paper work completely.

### **2.4.2 TECHNICAL FEASIBILITY**

The technical requirement for the system is economic and it does not use any other additional Hardware and software. Technical evaluation must also assess whether the existing systems can be upgraded to use the new technology and whether the organization has the expertise to use it.

### **2.4.3 OPERATIONAL FEASIBILITY**

The system working is quite easy to use and learn due to its simple but attractive interface. User requires no special training for operating the system. Technical performance includes issues such as determining whether the system can provide the right information for the Department personnel student details, and whether the system can be organized so that it always delivers this information at the right place and on time using intranet services. Acceptance revolves around the current system and its personnel.

## **CHAPTER 3**

### **SYSTEM SPECIFICATION**

#### **3.1 HARDWARE REQUIREMENTS (Minimum Requirement)**

- Minimum RAM: - 4GB
- Hard Disk: - 128 GB
- Processor: - Core 2 Duo or Higher Mother Board G41 (Gigabyte)

#### **3.2 SOFTWARE REQUIREMENTS (minimum Requirement)**

- Operating system: -Ubuntu 16.04
- Technology: -Python (Django Framework)
- Database: - MY-SQL 14.14
- IDE: - Sublime
- Browser: - Mozilla/Google Chrome/IE/Opera

#### **Functional & Non-Functional Requirement of software:**

##### **A) Functional requirements**

These are related to the expectations of the intended software. They describe what the software has to do. They are also called product features. Sometimes, functional requirements may also specify what the software should not do.

Functional Requirements should include:

- Main Page: It contains a single login form with a signup for the user. Users can log in to enter the system as a legit user.
- Form Page: It contains many fields such as roll no, subject, course, time, date, etc. Users can get the result after submitting the form.
- Face Recognition Feature: Students have to make sure that their faces are visible in the captured image.

## **B) Non-Functional requirements**

In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors.

This should be contrasted with functional requirements that define specific behavior or functions. The plan for implementing functional requirements is detailed in the system design.

The plan for implementing non-functional requirements is detailed in the system architecture.

- Practicality: The system is quite stable and can be operated by people with average intelligence.
- Efficiency: I tried to involve accuracy, timeliness, and comprehensiveness of the system output.
- Cost: It is desirable to aim for the system with a minimum cost subject to the condition that it must satisfy the entire requirement.
- Flexibility: I have tried that the system should be modifiable depending on the changing needs of the user. Such modifications should entail extensive reconstructing or recreation of software. It should also be portable to different computer systems.
- Security: This is a very important aspect that I followed in this designing phase and tried to cover the areas of hardware reliability, fall back procedures and physical security of data.

## **CHAPTER 4**

### **PROJECT DESCRIPTION**

#### **4.1 PROBLEM DEFINITION**

This system developed will reduce manual work and avoid redundant data. By maintaining the attendance manually, then accurate attendance cannot be marked. The system can mark accurate attendance. As the attendances are maintained in registers it has been a tough task for admin and staff to maintain for a long time. Instead, the software can keep long and retrieve the information when needed.

#### **4.2 PROJECT OVERVIEW**

Attendance System has three main modules for proper functioning:

- The admin module has rights for creating any new entry of teacher and student details.
- The teacher has the rights for creating any new entry of student details, student attendance, check attendance, create class time, check student profile, check their profile.
- The student has a right to mark attendance, check attendance, and profile check.

#### **4.3 MODULE DESCRIPTION**

The system should be designed in such a way that only authorized people should be allowed to access some particular modules. The records should be modified by only administrators and no one else. The user should always be in control of the application and not the vice versa.

The user interface should be consistent so that the user can handle the application with ease and speed. The application should be visually, conceptually clear.

##### **4.3.1 ADMINISTRATOR MODULE**

- Student Details:

This module deals with the allocation of roll no and details for a new batch. It will add detail of a student with the photos. It provides the student to have a user name and password.

- Teacher Details:

This module deals with the allocation of details for a new teacher. It will add details of a teacher with the photos. It provides the teacher to have a user name and password.

- Course Details:

This module deals with the addition of details of a course.

- Subject Details:

This module deals with the addition of details of a subject.

- Class Time Details:

This module deals with the addition of class time of a subject. Attendance can be marked for a class in time. It will help the admin, teacher, and student to make the entry of attendance based on the subject.

- Attendance details:

This module deals with the details of marked attendance. It will help the admin and teacher to analyze student's attendance.

#### **4.3.2 TEACHER MODULE**

- Student Details:

This module deals with the details of students.

- Class Time Details:

This module deals with the addition of class time of a subject. Attendance can be marked for a class in time. It will help the admin, teacher, and student to make the entry of attendance based on the subject.

- Attendance details:

This module deals with the details of marked attendance. It will help the admin and teacher to analyze student's attendance.

- Profile Details:

This module deals with the details of the teacher itself.

#### **4.3.3 STUDENT MODULE**

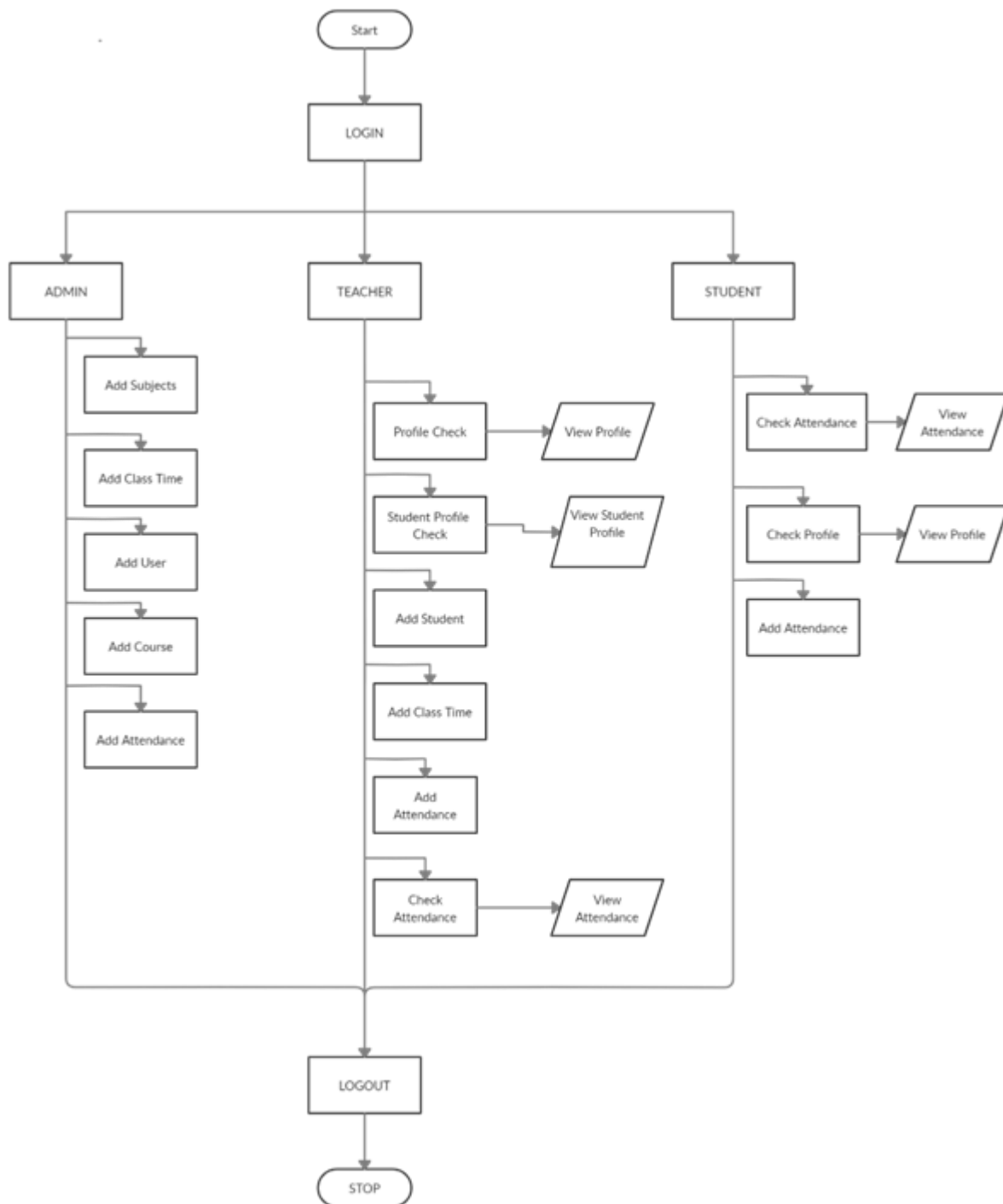
- Attendance details:

This module deals with the details of marked attendance. It will help the admin and teacher to analyze student's attendance.

- Profile Details:

This module deals with the details of the teacher itself.

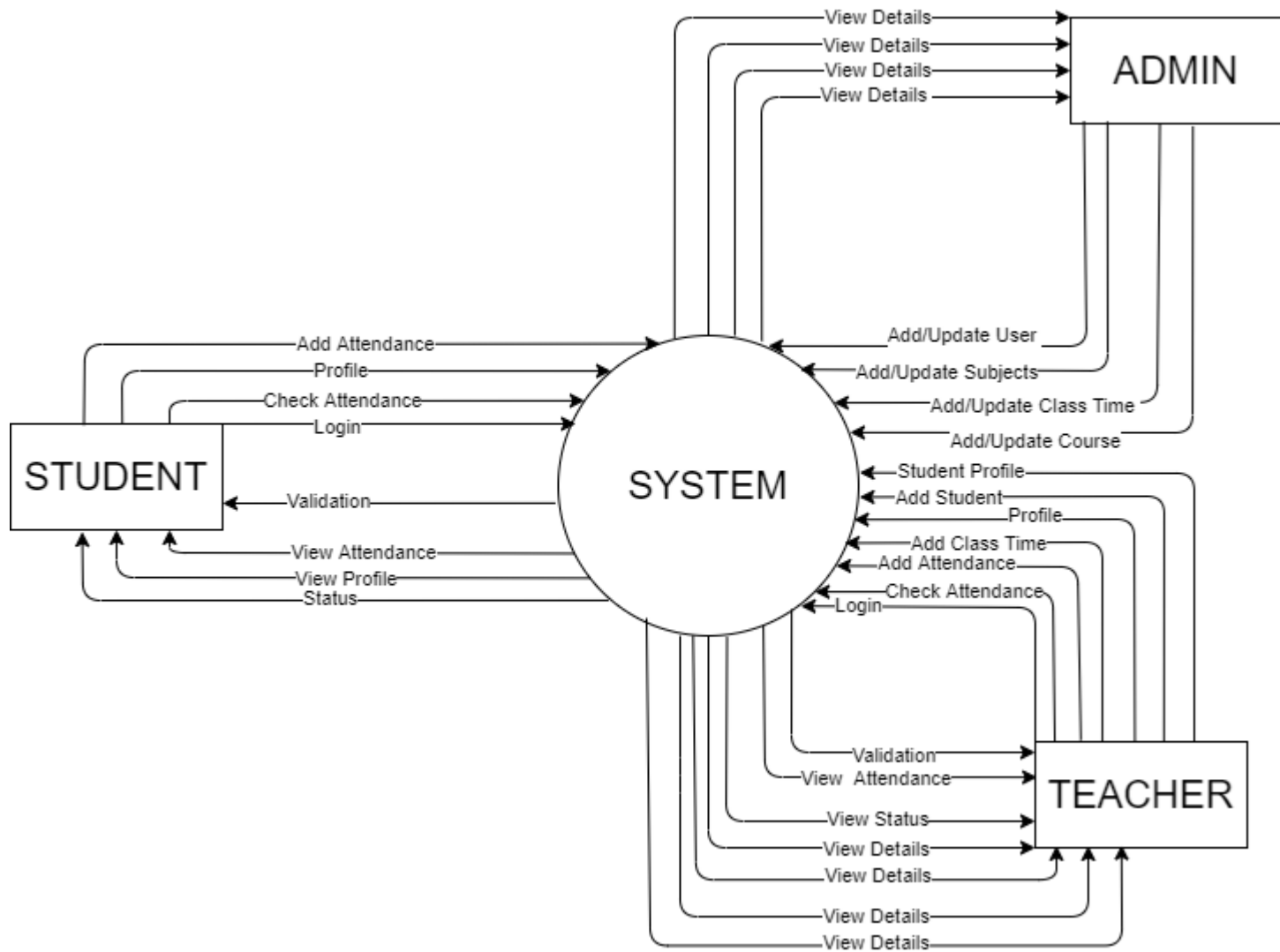
#### 4.4 SYSTEM FLOW DIAGRAM:





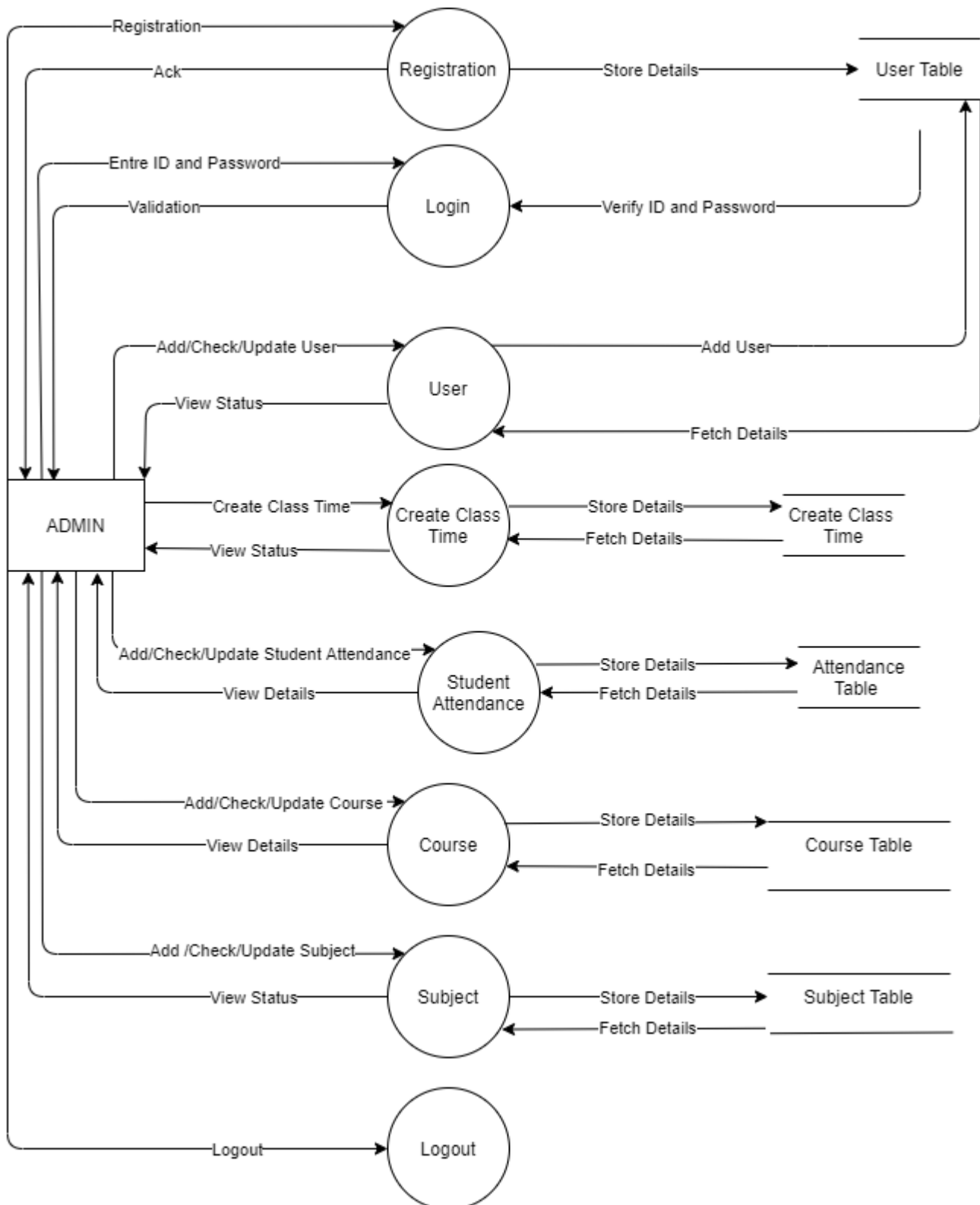
## 4.5 DATA FLOW DIAGRAM

### 4.5.1 DFD LEVEL 0

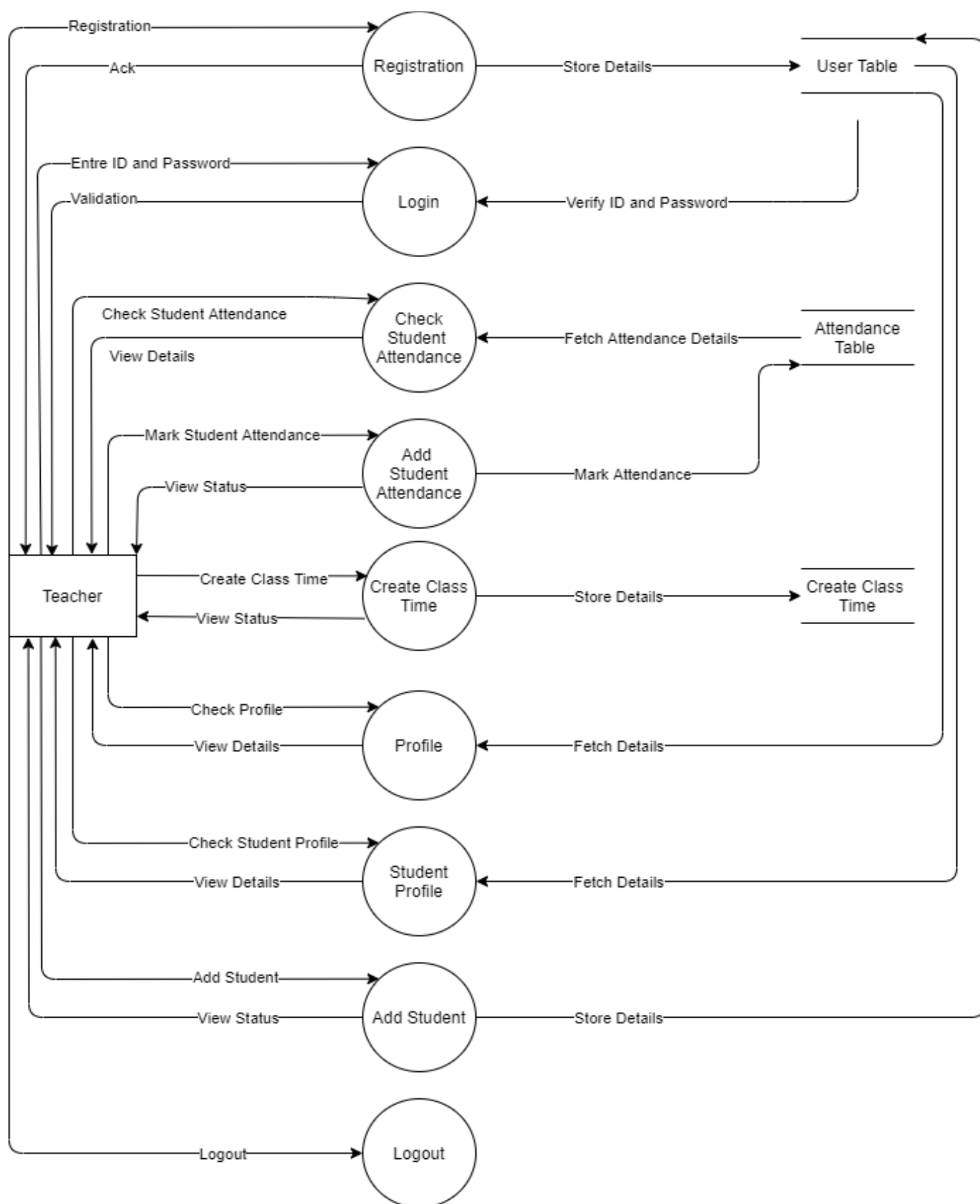


## 4.5.2 DFD LEVEL 1

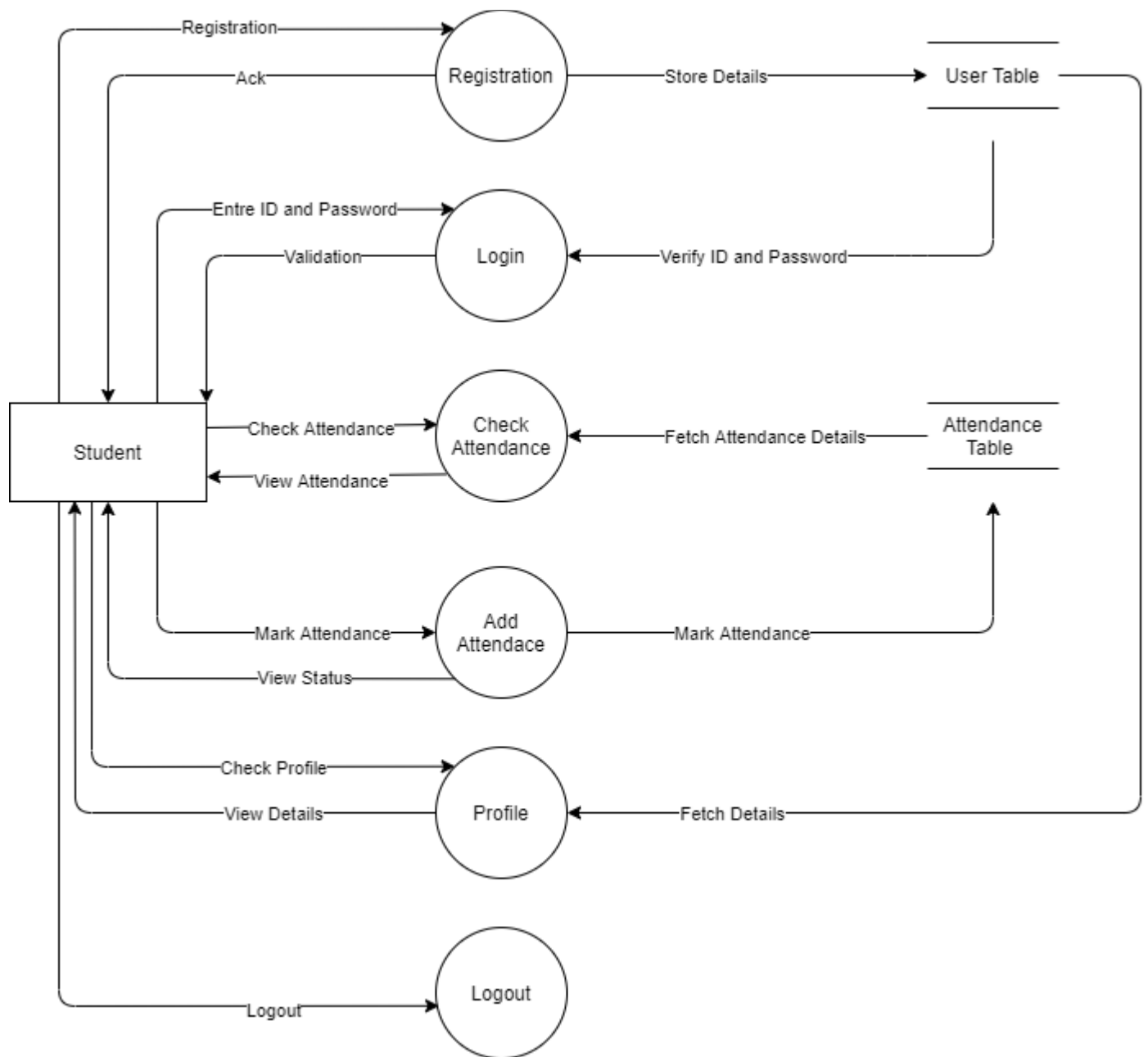
### 4.5.2.1 ADMIN DFD 1



#### 4.5.2.2 TEACHER DFD 1

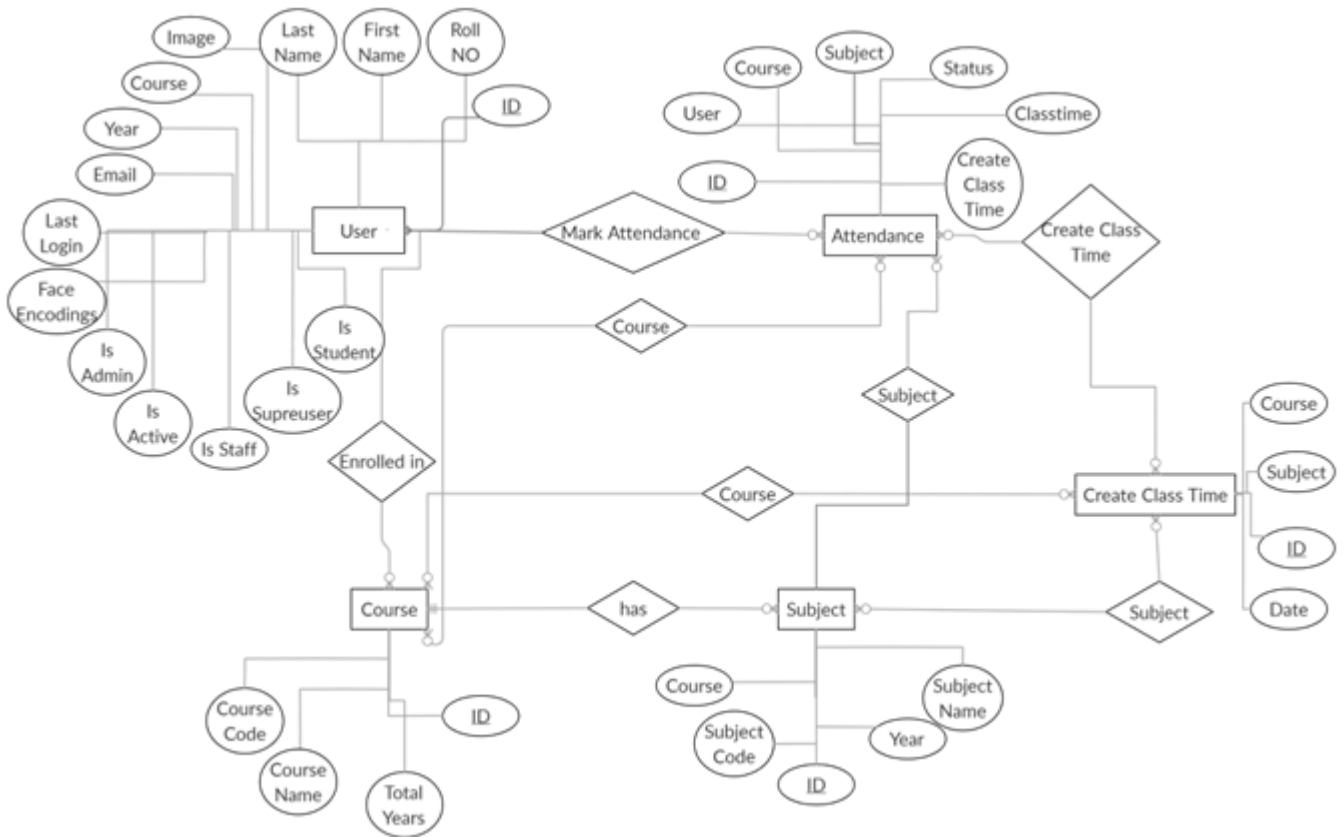


#### 4.5.2.3 STUDENT DFD 1

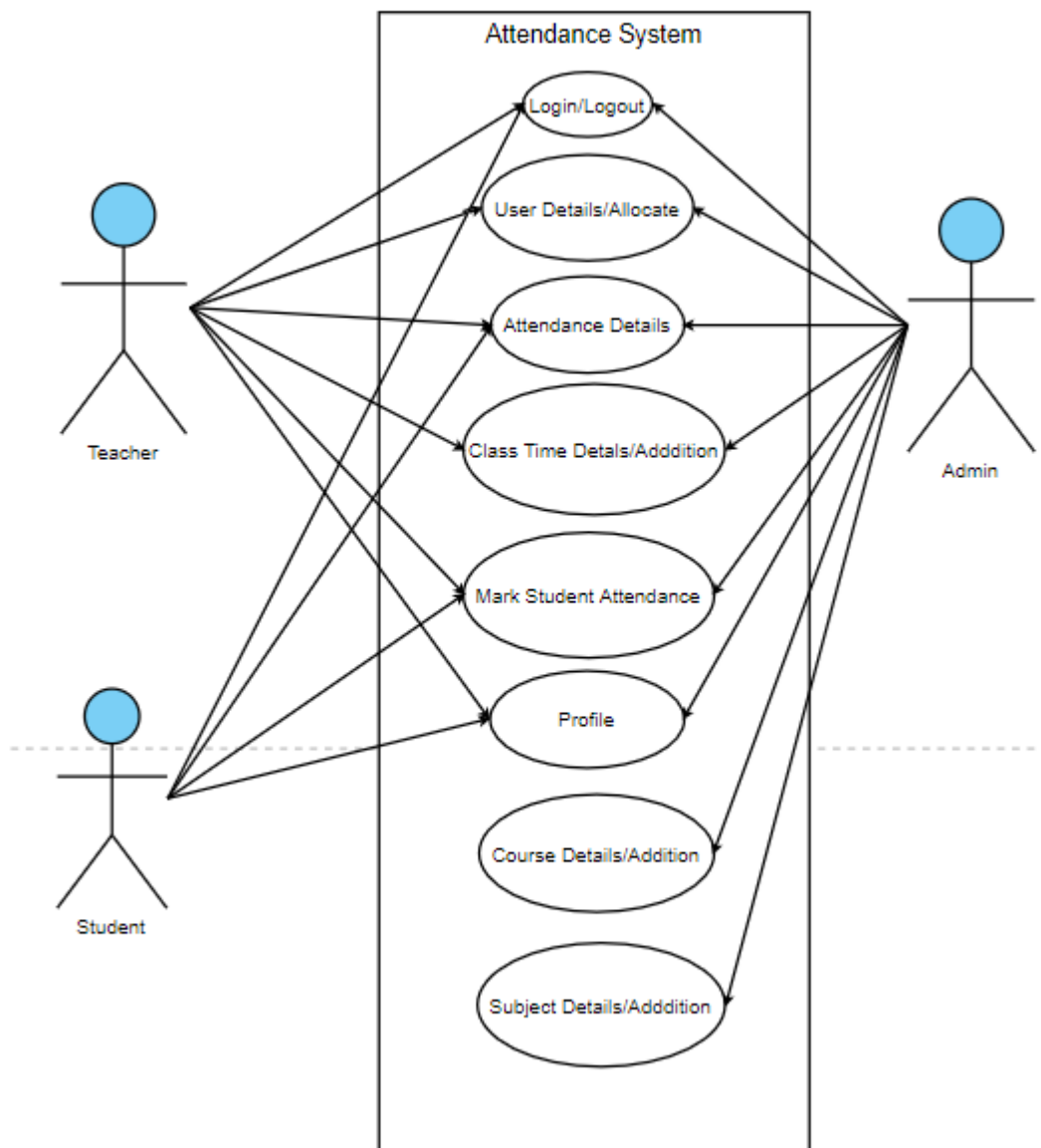


## 4.6 SYSTEM DESIGN

### 4.6.1 ENTITY RELATIONSHIP DIAGRAM



#### 4.6.2 USE CASE DIAGRAM



### 4.6.3 DATABASE DESIGN

#### 4.6.3.1 User Table

Feild	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto increament
password	varchar(128)	NO		NULL	
roll no	int(11)	NO		NULL	
first name	varchar(10)	NO		NULL	
last name	varchar(10)	NO		NULL	
year	varchar(10)	NO		NULL	
email	varchar(10)	NO	UNI	NULL	
date joined	datetime(6)	NO		NULL	
last login	datetime(6)	NO		NULL	
face encoding	varchar(100)	YES		NULL	
is admin	tinyint(1)	NO		NULL	
is_active	tinyint(1)	NO		NULL	
is staff	tinyint(1)	NO		NULL	
is superuser	tinyint(1)	NO		NULL	
is student	tinyint(1)	NO		NULL	
course id	tinyint(1)	NO	MUL	NULL	
img	varchar(100)	NO		NULL	

#### 4.6.3.2 ATTENDANCE TABLE

Feild	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto increament
status	tinyint(1)	NO		NULL	
subject id	int(11)	NO	MUL	NULL	
user id	int(11)	NO	MUL	NULL	
classtime	datetime(6)	YES		NULL	
course id	int(11)	YES	MUL	NULL	
actualclasstime id	int(11)	YES	MUL	NULL	

#### 4.6.3.3 COURSE TABLE

Feild	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto increament
code	varchar(10)	NO	UNI	NULL	
c name	varchar(10)	NO	UNI	NULL	
total years	varchar(2)	NO		NULL	

#### 4.6.3.4 SUBJECT TABLE

Feild	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto increament
s name	varchar(10)	NO		NULL	
s code	varchar(10)	NO		NULL	
years	varchar(2)	NO		NULL	
course id	int(11)	NO	MUL	NULL	

#### 4.6.2.5 CREATE CLASS TIMETABLE

Feild	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto increament
date	datetime(6)	NO		NULL	
course id	int(11)	NO	MUL	NULL	
subject id	int(11)	NO	MUL	NULL	



### **4.6.3 INPUT DESIGN**

Input design is part of overall system design that requires special attention designing input data is to make the data entered easy and free from errors.

Input design is the process of converting the user originated inputs to a computer-based format. A system user interacting through a workstation must be able to tell the system whether to accept the input to result. The collection of input data is considered to be the most expensive part of the system design. Since the input has to be planned in such a manner to get relevant information, extreme care is taken to obtain pertinent information.

You need to enter login details first to get into the system.

Students have to use their faces to mark attendance. Teachers can also mark the attendance by just filling details.

Students can check their attendance, profile, and mark their attendance by filling the form. The teachers can also check student attendance, student profile, profile, and new student by filling the form. Input should be valid.

### **4.6.4 OUTPUT DESIGN**

Output design for this application “Face Recognition based Attendance system” generally refers to the results and information that are generated by the system for many end-users; output is the main reason for developing the system and the basis on which they evaluate the usefulness of the application. The output is designed in such a way that it is attractive, convenient, and informative. Forms are designed with various features, which make the console output more pleasing. As the outputs are the most important sources of information to the users, the better design should improve the system’s relationships with us and also will help in decision making. Form design elaborates on the way output is presented and the layout available for capturing information. One of the most important factors of the system is the output it produces. This system refers to the results and information generated. Basically the output from a computer system is used to communicate the result of processing to the user. Attendance system to show the report subject wise attendance maintaining by the teacher. Taken as a whole report obtain on administrator privileges only. These forms will show reports based on a roll no, subject, and course to our end user. As the outputs are the most important sources of information to the users, the better design should improve the system’s relationships with us and also will help in decision making. Form design elaborates on the way output is presented and the layout available for capturing information.

## **CHAPTER 5**

### **SYSTEM TESTING**

#### **5.1 INTRODUCTION**

Once source code has been generated, software must be tested to uncover (and correct) as many errors as possible before delivery to customer. Our goal is to design a series of test cases that have a high likelihood of finding errors. To uncover the errors software techniques are used. These techniques provide systematic guidance for designing test that

- (1) Exercise the internal logic of software components, and
- (2) Exercise the input and output domains of the program to uncover errors.

In program function, behavior and performance.

##### **5.1.1 Steps:**

**Software is tested from two different perspectives:**

- (1) Internal program logic is exercised using —White box test case design Techniques.
- (2) Software requirements are exercised using —block box test case Design techniques.

In both cases, the intent is to find the maximum number of errors with the Minimum amount of effort and time.

#### **5.2 TESTING METHODOLOGIES**

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements. A strategy must provide guidance for the practitioner and a set of milestones for the manager. Because the steps of the test strategy occur at a time when deadline pressure begins to rise, progress must be measurable and problems must surface as early as possible. Following testing techniques are well known and the same strategy is adopted during this project testing.

##### **5.2.1 UNIT TESTING**

Unit testing focuses on verification efforts on the smallest unit of software design the software component or module. The unit test is white-box oriented. The unit testing is implemented in every module of the Student Attendance System. by giving correct manual input to the system, the data are stored in the database and retrieved. If you want the required module to access input or get the output from the End-user. any error will accrue the time will provide a handler to show what type of error will be accrued.

### **5.2.2 INTEGRATION TESTING**

The objective is to take unit tested components and build a program structure that has been dictated by design. Integration testing is testing in which a group of components are combined to produce output. Integration testing is of four types: (i) Top down (ii) Bottom up (iii) Sandwich (iv) Big-Bang

### **5.2.3 SYSTEM TESTING**

System testing is a series of different tests whose primary purpose is to fully exercise the computer-based system. If you want to change any values or inputs will change all information. so specified input is a must.

### **5.2.4 DEFECT TESTING**

- The goal of defect testing is to discover defects in programs.
- A successful defect test is a test which causes a program to behave in an anomalous way.
- Tests show the presence not the absence of defects.

### **5.2.5 BLOCK BOX TESTING**

- An approach to testing where the program is considered as a 'black-box'.
- The program test cases are based on the system specification.
- Test planning can begin early in the software process

## 5.3 TEST CASES

Test case is an object for execution for other modules in the architecture does not represent any interaction by itself. A test case is a set of sequential steps to execute a test operating on a set of predefined inputs to produce certain expected outputs.

There are two types of test cases: -manual and automated. A manual test case is executed manually while an automated test case is executed using automation.

In system testing, test data should cover the possible values of each parameter based on the requirements. Since testing every value is impractical, a few values should be chosen from each equivalence class. An equivalence class is a set of values that should all be treated the same. Ideally, test cases that check error conditions are written separately from the functional test cases and should have steps to verify the error messages and logs. Realistically, if functional test cases are not yet written, it is ok for testers to check for error conditions when performing normal functional test cases. It should be clear which test data, if any is expected to trigger errors.

### **White Box Test Cases:**

- Guarantee that all independent paths within a module have been exercised at least once.
- Exercise all logical decisions on their true and false sides.
- Execute all loops at their boundaries and within their operational bounds.
- Exercise internal data structures to ensure their validity.

### **Black Box Test Cases:**

- Incorrect or missing functions,
- Interface errors,
- Errors in data structures or external data base access.
- Behavior or performance errors,
- Initialization and termination errors.
- Test cases that reduce the number of additional test cases that must be designed to achieve reasonable testing (i.e minimize effort and time).
- Test cases that tell us something about the presence or absence of classes of errors

## **CHAPTER 6**

### **SYSTEM IMPLEMENTATION**

#### **6.1 PURPOSE**

System implementation is the important stage of project when the theoretical design is tuned into practical system. The main stages in the implementation are as follows:

- Planning
- Training
- System testing and
- Changeover Planning

Planning is the first task in the system implementation. At the time of implementation of any system people from different departments and system analysis involve. They are confirmed to practical problem of controlling various activities of people outside their own data processing departments.

The line managers controlled through an implementation coordinating committee. The committee considers ideas, problems and complaints of user department, it must also consider:

- The implication of system environment
- Self-selection and allocation for implementation tasks
- Consultation with unions and resources available
- Standby facilities and channels of communication

#### **IMPLEMENTATION**

Student Attendance system will implement student details, teachers' detail, subjects' details, separate login details, class time details, course details, and attendance details. This system will provide attendance based on subjects, roll no, and courses selected by the end-user.

Students will be able to check attendance based on subject and mark attendance based on available class time. The teachers will be able to check attendance, mark attendance, create class time, check their profile, check students' profiles, and adding new students. Admin will have all the option that students and teachers have.

The face recognition feature is based on Convolutional Neural Network. Students have to provide at least 4 to 5 images to make sure their face is captured from different angles. These 4 to 5 images will be converted into numerical values that dimension is 128d. We have used a pre-trained model to convert images into 128-dimensional numerical values. When students mark attendance, their face is again encoded into 128-d and matches against the encoded values(4 to 5 images). If Mark time is in-class time and faces matches, then the attendance will be marked.

## **6.2 SYSTEM MAINTENANCE**

Software maintenance is far more than finding mistakes. Provision must be made for environment changes, which may affect either the computer, or other parts of the computer-based systems. Such activity is normally called maintenance. It includes both the improvement of the system functions and the corrections of faults, which arise during the operation of a new system.

It may involve the continuing involvement of a large proportion of computer department resources. The main task may be to adapt existing systems in a changing environment.

Back up for the entire database files are taken and stored in storage devices like flash drives, pen drives and disks so that it is possible to restore the system at the earliest. If there is a breakdown or collapse, then the system gives provision to restore database files. Storing data in a separate secondary device leads to an effective and efficient maintenance of the system. The nominated person has sufficient knowledge of the organization's computer based system to be able to judge the relevance of each proposed change.

## **CHAPTER 7**

### **CONCLUSION AND FUTURE ENHANCEMENT**

#### **7.1 CONCLUSION**

To conclude, Project Data Grid works like a component that can access all the databases and picks up different functions. Students and Teachers will have a dashboard to do operations. Students can mark attendance, check attendance, and check their profile. The teacher can mark students' attendance, check attendance, check their profile, create a class time, and adding new students. Admin can add/update student, courses, subjects, teachers, and attendance.

It overcomes the many limitations incorporated in the attendance.

- Easy implementation Environment
- Generate results Flexibly.
- East to use.

#### **7.2 SCOPE FOR FUTURE DEVELOPMENT**

The project has a very vast scope in the future. The project can be updated in the near future as and when the requirement for the same arises, as it is very flexible in terms of expansion.

- It can be updated in the Management System.
- Teacher Attendance System can be integrated with it.
- Biometric Technology can be used in the future.
- A short Attendance report feature will be added.
- Automatically short Attendance report will send to the teacher and parents of students.

## CHAPTER 8

### APPENDICES

#### 8.1 SOURCE CODE

##### Settings.py

```
import os

# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
AUTHENTICATION_BACKENDS = (
    ('django.contrib.auth.backends.ModelBackend'),
)

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/2.2/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'f(th7o-qy0a-c6yrk40m*9837j*os-zr^ijst%tbxs6qho!vx@'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = ['*']

# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'studentface',
    'django.forms',
]
```



```

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    # 'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'attendance_system.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [BASE_DIR + '/templates/'],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    ],
]

LOGIN_REDIRECT_URL = 'Stud_dashboard'
LOGOUT_REDIRECT_URL = 'index'
WSGI_APPLICATION = 'attendance_system.wsgi.application'

```

```

# Database
# https://docs.djangoproject.com/en/2.2/ref/settings/#databases

```

```

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'db_attendence',
        'USER': 'root',
        'PASSWORD': 'admin',
        'HOST': 'localhost',
        'PORT': '3306',
    }
}

```

```

}
ROOT_URLCONF = 'attendance_system.urls'
WSGI_APPLICATION = 'attendance_system.wsgi.application'

# Password validation
# https://docs.djangoproject.com/en/2.2/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]

AUTH_USER_MODEL = 'studentface.User'
# Internationalization
# https://docs.djangoproject.com/en/2.2/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'Asia/Kolkata'

USE_I18N = True

USE_L10N = True

USE_TZ = False

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR, 'templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',

```

```

        'django.template.context_processors.request',
        'django.contrib.auth.context_processors.auth',
        'django.contrib.messages.context_processors.messages',
    ],
},
},
]

```

```

# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/2.2/howto/static-files/
PROJECT_ROOT = os.path.dirname(os.path.abspath(__file__))
STATIC_URL = '/static/'
STATIC_ROOT = os.path.join(PROJECT_ROOT, 'static')

```

```

STATICFILES_DIRS = [os.path.join(BASE_DIR, 'static')]

```

```

MEDIA_ROOT = os.path.join(BASE_DIR, 'media')

```

```

MEDIA_URL = '/media/'

```

```

FORM_RENDERER = 'django.forms.renderers.TemplatesSetting'

```

```

from django.contrib.messages import constants as messages

```

```

MESSAGE_TAGS = {
    messages.DEBUG: 'alert-secondary',
    messages.INFO: 'alert-info',
    messages.SUCCESS: 'alert-success',
    messages.WARNING: 'alert-warning',
    messages.ERROR: 'alert-danger',
}

```

### **Urls.py**

```

from django.urls import path , include
from django.conf import settings
from django.conf.urls.static import static
from .views import *
from django.contrib.auth import views as auth_views

```

```

urlpatterns = [

    path("",index.as_view(), name="index"),
    path('stud_att',stud_att, name="stud_att"),

```

```

path('att_check',att_check,name="att_check"),

path('clickimg',clickimg,name='clickimg'),
path('upload_img',upload_img,name='upload_img'),
path('Registerstudent',Registerstudent,name='Registerstudent'),
path('Registerteacher',Registerteacher,name='Registerteacher'),
path('login',login,name='login'),
path('signuppagestudent',signuppagestudent,name='signuppagestudent'),
path('signuppageteacher',signuppageteacher,name='signuppageteacher'),
path('add_user_teacher',add_user_teacher,name='add_user_teacher'),
path('add_user_student',add_user_student,name='add_user_student'),
path('login/', auth_views.LoginView.as_view(), name='login'),
path('logout/', auth_views.LogoutView.as_view(), name='logout'),
path('Stud_dashboard/',Stud_dashboard , name='Stud_dashboard'),
path('Teach_dashboard/',Teach_dashboard , name='Teach_dashboard'),
path('createclass_time/',createclass_time, name='createclass_time'),
path('addclasstime/',addclasstime, name='addclasstime'),
path('classtime/',classtime, name='classtime'),
path('checkattendance/',checkattendance, name='checkattendance'),
path('check/',check, name='check'),

path('showattbydate/',showattbydate, name='showattbydate'),
path('add_att_by_teacher/',add_att_by_teacher, name='add_att_by_teacher'),

path('<int:pk>/',edituserprofile, name='edituserprofile'),

path('studprofile/',studprofile,name='studprofile'),
path('studprofilecheck/',studprofilecheck,name='studprofilecheck'),

path('editprofilebyteach',editprofilebyteach,name='editprofilebyteach'),
path('addstudent',addstudent,name='addstudent'),
path('trainstudface/<int:pk>/',trainstudface,name='trainstudface')

]
if settings.DEBUG: # new
    urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)

```

## Models.py

```

from django.db import models
import datetime
# Create your models here.
from django.contrib.auth.models import AbstractBaseUser , BaseUserManager

```

```
YEARS = (
    ('1', 1),
    ('2', 2),
    ('3', 3),
    ('4', 4),
    ('5', 5),
    ('NA','NA')
)
```

```
CATEGORY_CHOICES2 = (
    ('0', 0),
    ('1', 1),
    ('2', 2),
)
```

```
class studentManage(BaseUserManager):
    def create_user(self,email,first_name,password=None):
        if not email:
            raise ValueError("users must have an email")

        if not first_name:
            raise ValueError("users must have an first_name")
        user = self.model(
            email=self.normalize_email(email),
            first_name = first_name,
            roll_no = 00,
        )
        user.set_password(password)
        user.save(using=self._db)
        return user

    def create_superuser(self,email,first_name,password):
        user=self.create_user(
            email=self.normalize_email(email),
            first_name =first_name,
            password = password
        )
        user.is_admin=True
        user.is_staff=True
        user.is_superuser=True
        user.save(using=self._db)
```

```
return user
```

```
class course(models.Model):
```

```
    #roll_no=models.OneToOne
```

```
    code=models.CharField(max_length=10, unique=True)
```

```
    c_name=models.CharField(max_length=10,unique=True)
```

```
    total_years=models.CharField(choices=YEARS, max_length=2)
```

```
    def __str__(self):
```

```
        return self.c_name
```

```
class User(AbstractBaseUser):
```

```
    roll_no= models.IntegerField(default='000')
```

```
    img = models.ImageField(upload_to='file/',default='file/S.jpg')
```

```
    #username=models.CharField(max_length=10,unique=True)
```

```
    first_name=models.CharField(max_length=10)
```

```
    last_name=models.CharField(max_length=10)
```

```
    course=models.ForeignKey(course,on_delete=models.CASCADE,null=True,blank=True)
```

```
    year=models.CharField(choices=YEARS, max_length=2,default='NA')
```

```
    email=models.EmailField(verbose_name='email', max_length=20, unique=True)
```

```
    date_joined=models.DateTimeField(verbose_name='date joined', auto_now_add=True)
```

```
    last_login=models.DateTimeField(verbose_name='last login', auto_now=True)
```

```
    face_encoding=models.FileField(upload_to='file/',default='not applicable', null=True, blank=True)
```

```
    #img=models.ImageField(upload_to='image/',default="")
```

```
    is_admin=models.BooleanField(default=False)
```

```
    is_active=models.BooleanField(default=True)
```

```
    is_staff=models.BooleanField(default=False)
```

```
    is_superuser=models.BooleanField(default=False)
```

```
    is_student=models.BooleanField(default=False)
```

```
    USERNAME_FIELD= 'email'
```

```
    REQUIRED_FIELDS=['first_name',]
```

```
objects= studentManage()
```

```
def __str__(self):
```

```
    return str(self.roll_no)
```

```
def has_perm(self,perm,obj=None):
```

```
    return self.is_admin
```

```
def has_module_perms(self, app_label):
```

```
    return True
```

```
class subject(models.Model):
```

```

course=models.ForeignKey(course,on_delete=models.CASCADE)
s_name=models.CharField(max_length=10)
s_code=models.CharField(max_length=10)
year=models.CharField(choices=YEARS, max_length=2)

def __str__(self):
    return self.s_name

class createclasstime(models.Model):
    course=models.ForeignKey(course,on_delete=models.CASCADE)
    subject=models.ForeignKey(subject,on_delete=models.CASCADE)
    date=models.DateTimeField(verbose_name='date')
    def __str__(self):
        return '{ } { } { }'.format(self.course, self.subject,self.date)

class Attendance(models.Model):
    user=models.ForeignKey(User,on_delete=models.CASCADE)
    course=models.ForeignKey(course,on_delete=models.CASCADE,null=True)
    subject=models.ForeignKey(subject,on_delete=models.CASCADE)
    status=models.BooleanField(default=False)
    classtime=models.DateTimeField(null=True)
    actualclasstime=models.ForeignKey(createclasstime,on_delete=models.CASCADE,null=True)

```

## Index.html

```

{ % load static % }
<!DOCTYPE html>
<html lang="en">
<head>
<link rel="stylesheet" href="{ % static 'indexstyle.css' % }" >
<style>
.button {
display: inline-flex;
height: 40px;
width: 150px;
border: 2px solid black;
margin: 20px 20px 20px 20px;
color: #BFC0C0;
text-transform: uppercase;
text-decoration: none;
font-size: .8em;
letter-spacing: 1.5px;

```

```

align-items: center;
justify-content: center;
overflow: hidden;
}
@import url('https://fonts.googleapis.com/css?family=Exo:400,700');

*{
  margin: 0px;
  padding: 0px;
}

body{
  font-family: 'Exo', sans-serif;
}

.context {
  width: 100%;
  position: absolute;
  top: 50vh;
}

.context h1 {
  text-align: center;
  color: #fff;
  font-size: 50px;
}

.area{
  background: #e5dfdf;
  background: -webkit-linear-gradient(to left, #8f94fb, #4e54c8);
  width: 100%;
  height: 100vh;
}

.circles{
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
}

```



```
    overflow: hidden;
}
```

```
.circles li{
    position: absolute;
    display: block;
    list-style: none;
    width: 20px;
    height: 20px;
    background: rgba(255, 255, 255, 0.2);
    animation: animate 25s linear infinite;
    bottom: -150px;
}
```

```
.circles li:nth-child(1){
    left: 25%;
    width: 80px;
    height: 80px;
    animation-delay: 0s;
}
```

```
.circles li:nth-child(2){
    left: 10%;
    width: 20px;
    height: 20px;
    animation-delay: 2s;
    animation-duration: 12s;
}
```

```
.circles li:nth-child(3){
    left: 70%;
    width: 20px;
    height: 20px;
    animation-delay: 4s;
}
```

```
.circles li:nth-child(4){
    left: 40%;
    width: 60px;
    height: 60px;
    animation-delay: 0s;
    animation-duration: 18s;
}
```

```
.circles li:nth-child(5){  
  left: 65%;  
  width: 20px;  
  height: 20px;  
  animation-delay: 0s;  
}
```

```
.circles li:nth-child(6){  
  left: 75%;  
  width: 110px;  
  height: 110px;  
  animation-delay: 3s;  
}
```

```
.circles li:nth-child(7){  
  left: 35%;  
  width: 150px;  
  height: 150px;  
  animation-delay: 7s;  
}
```

```
.circles li:nth-child(8){  
  left: 50%;  
  width: 25px;  
  height: 25px;  
  animation-delay: 15s;  
  animation-duration: 45s;  
}
```

```
.circles li:nth-child(9){  
  left: 20%;  
  width: 15px;  
  height: 15px;  
  animation-delay: 2s;  
  animation-duration: 35s;  
}
```

```
.circles li:nth-child(10){  
  left: 85%;  
  width: 150px;  
  height: 150px;  
  animation-delay: 0s;  
  animation-duration: 11s;  
}
```

```

@keyframes animate {

    0%{
        transform: translateY(0) rotate(0deg);
        opacity: 1;
        border-radius: 0;
    }

    100%{
        transform: translateY(-1000px) rotate(720deg);
        opacity: 0;
        border-radius: 50%;
    }

}

```

```

.wrapper {
margin-right: auto; /* 1 */
margin-left: auto; /* 1 */

max-width: 714px; /* 2 */

padding-right: 10px; /* 3 */
padding-left: 10px; /* 3 */
}
a{ font-size:20px;
color:black;
text-align:right;
}
p{ font-size:19px;
color: black;
}

```

```

</style>

```

```

</head>

```

```

<body>

```

```

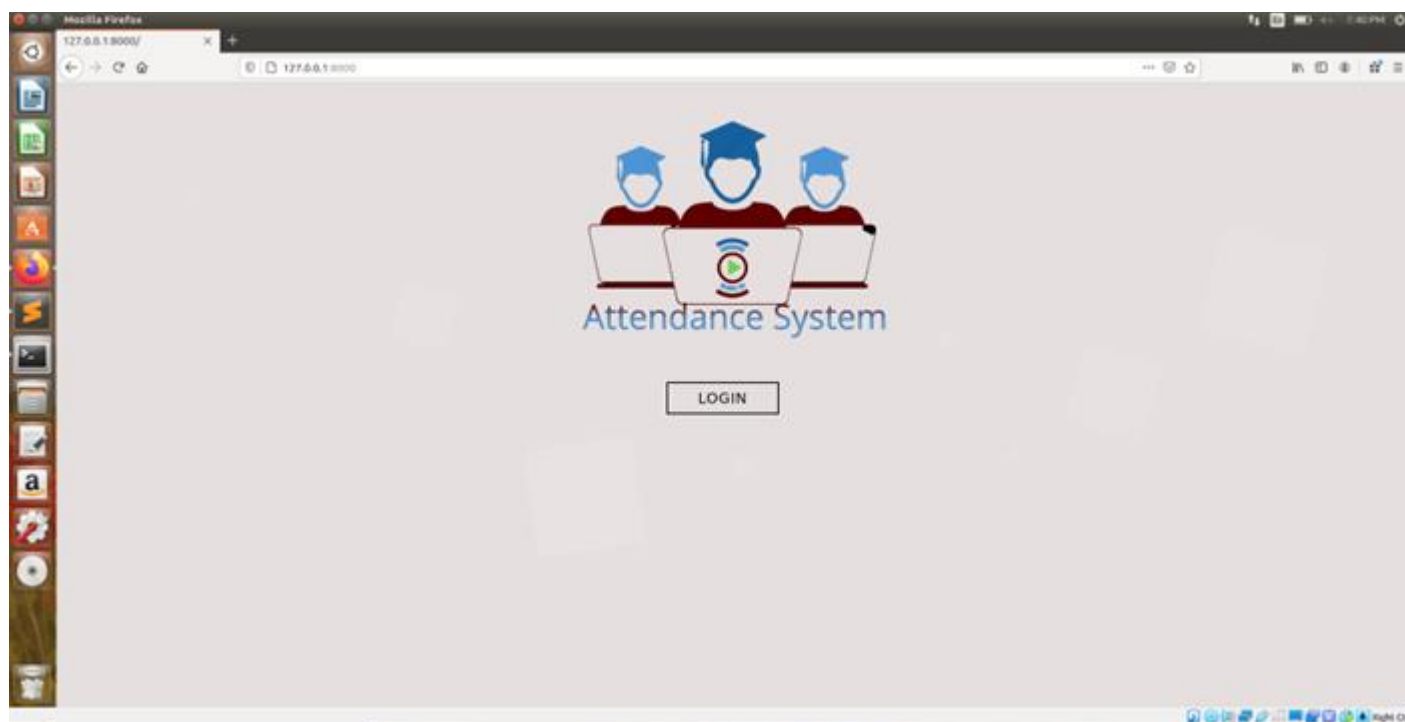
<div class="area" >
    <ul class="circles">
        <li></li>
        <li></li>
        <li></li>
    </ul>

```

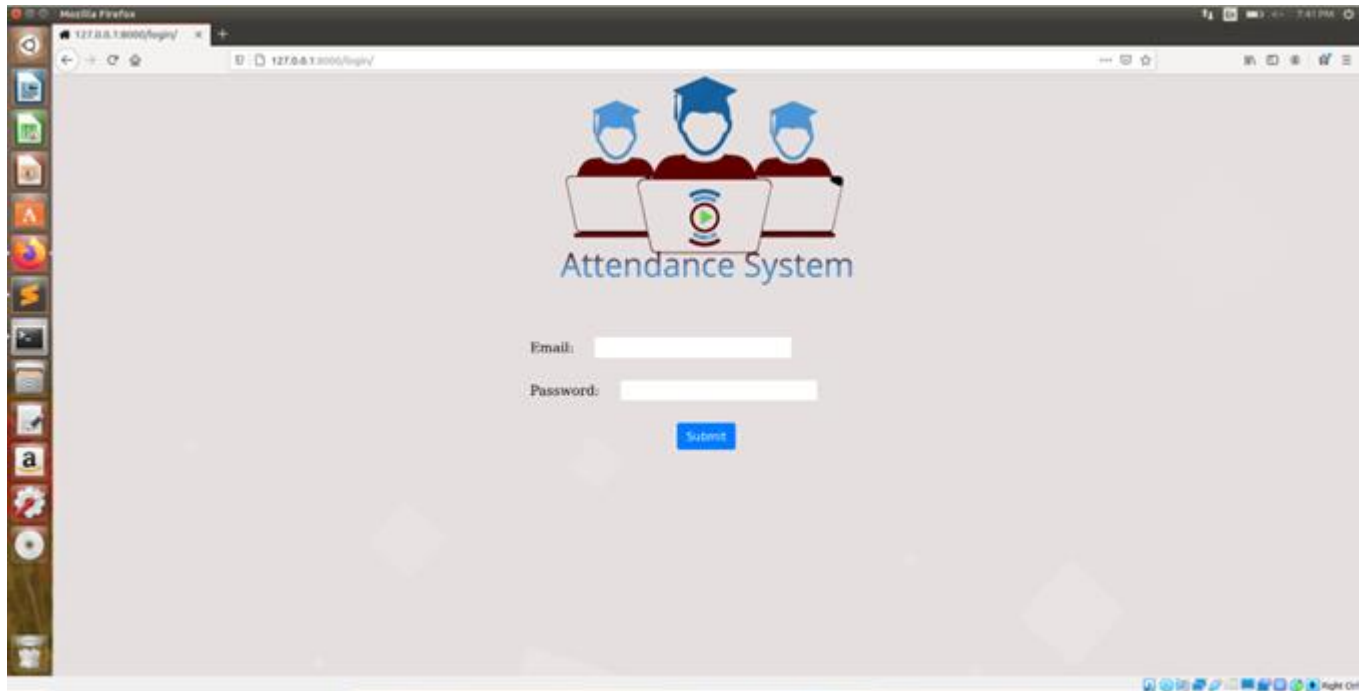
&lt;li&gt;&lt;/li&gt;

## 8.2 SCREEN SHOTS

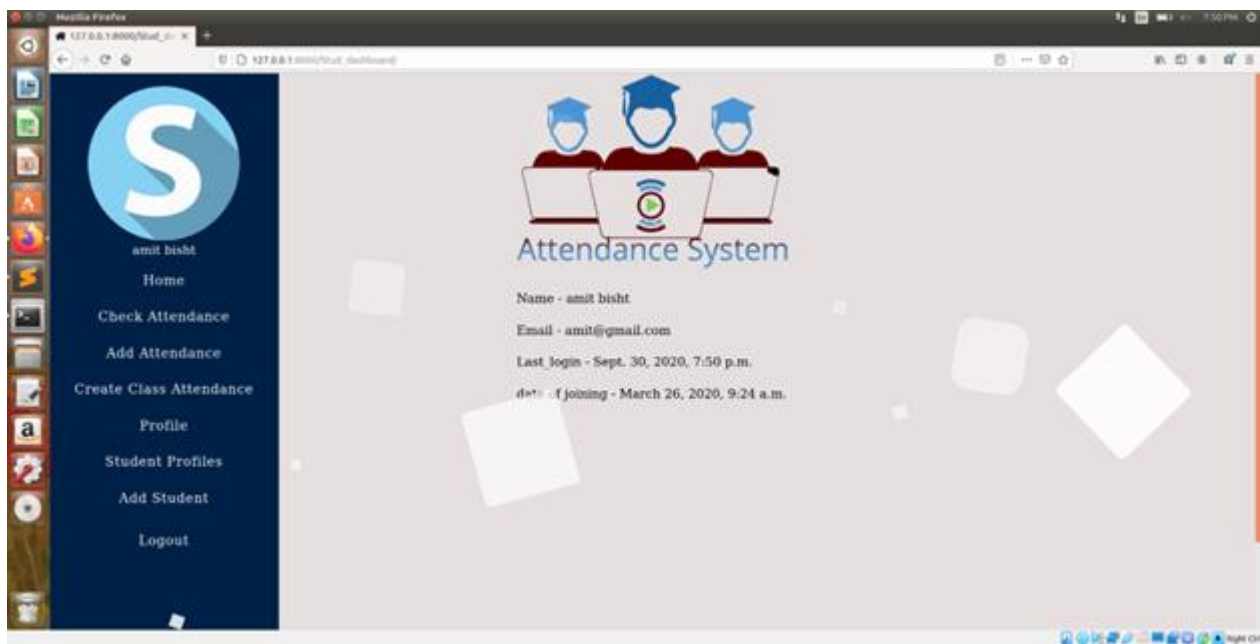
### Home Screen



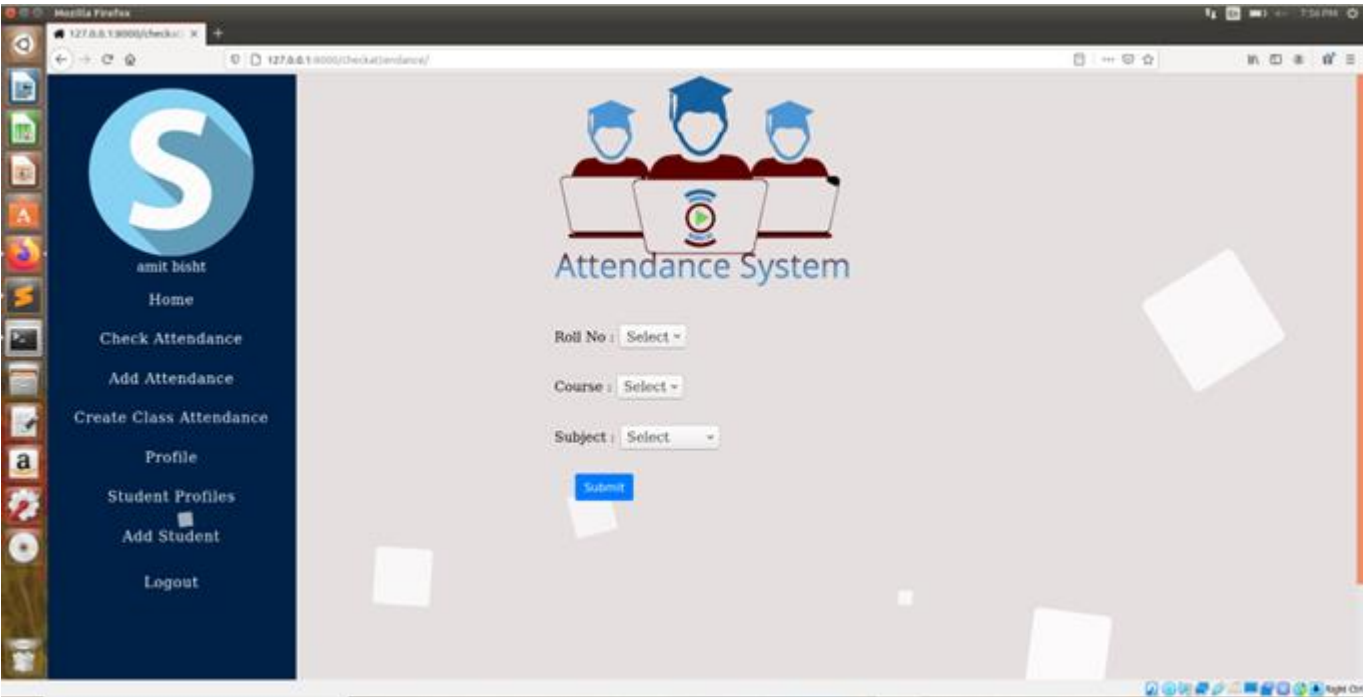
## Login:



## Teacher dashboard:



Teacher Check Attendance

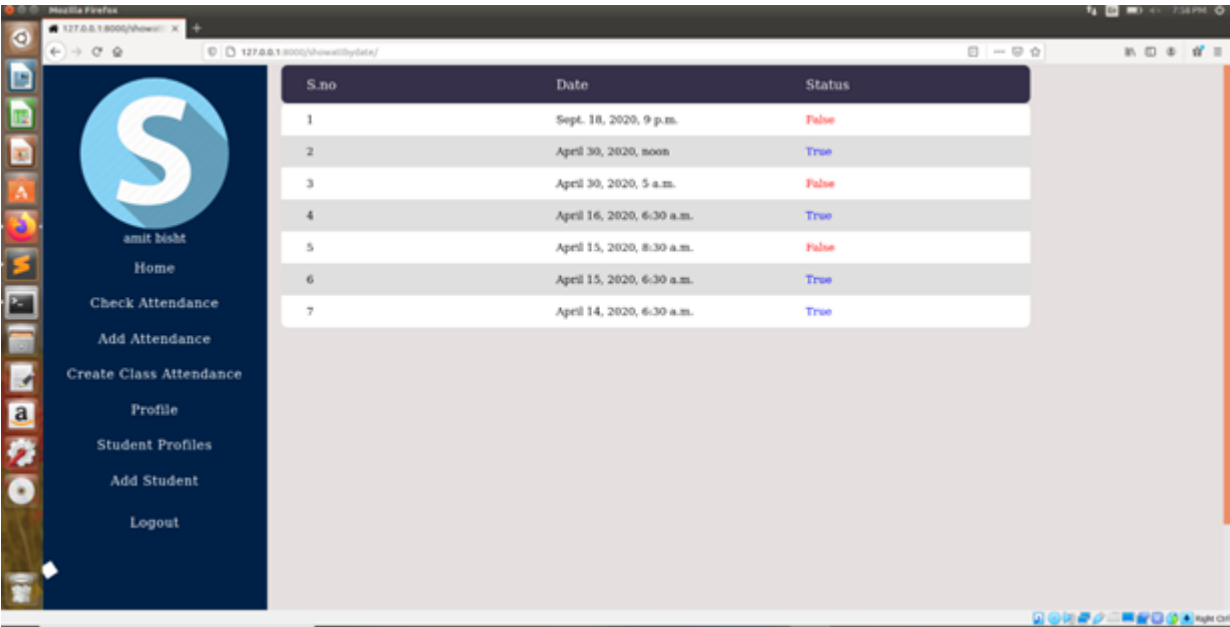


Teacher Check Attendance Table

The screenshot shows the same web application as before, but with a table of attendance records displayed in the main content area. The table has seven columns: 'S.no', 'Roll No', 'Student Name', 'Course Name', 'Subject Name', 'Class Attended', and 'By Date'. The table contains 12 rows of data. The sidebar and header remain the same as in the previous screenshot.

S.no	Roll No	Student Name	Course Name	Subject Name	Class Attended	By Date
1	7	Mahesh Bisht	BVOC	C++	4	Submit
2	7	Mahesh Bisht	BVOC	Networking	2	Submit
3	7	Mahesh Bisht	BVOC	C	1	Submit
4	1	Robannnnnn Singh	BCA	JAVA	1	Submit
5	4	Ashish Kumar	BVOC	C++		Submit
6	4	Ashish Kumar	BVOC	Networking		Submit
7	4	Ashish Kumar	BVOC	C		Submit
8	10	akash kumar	BCA	JAVA		Submit
9	3	Manish Bisht	BVOC	C	1	Submit
10	3	Manish Bisht	BVOC	C++		Submit
11	3	Manish Bisht	BVOC	Networking		Submit
12	11	Shubham Singh	BCA	JAVA		Submit

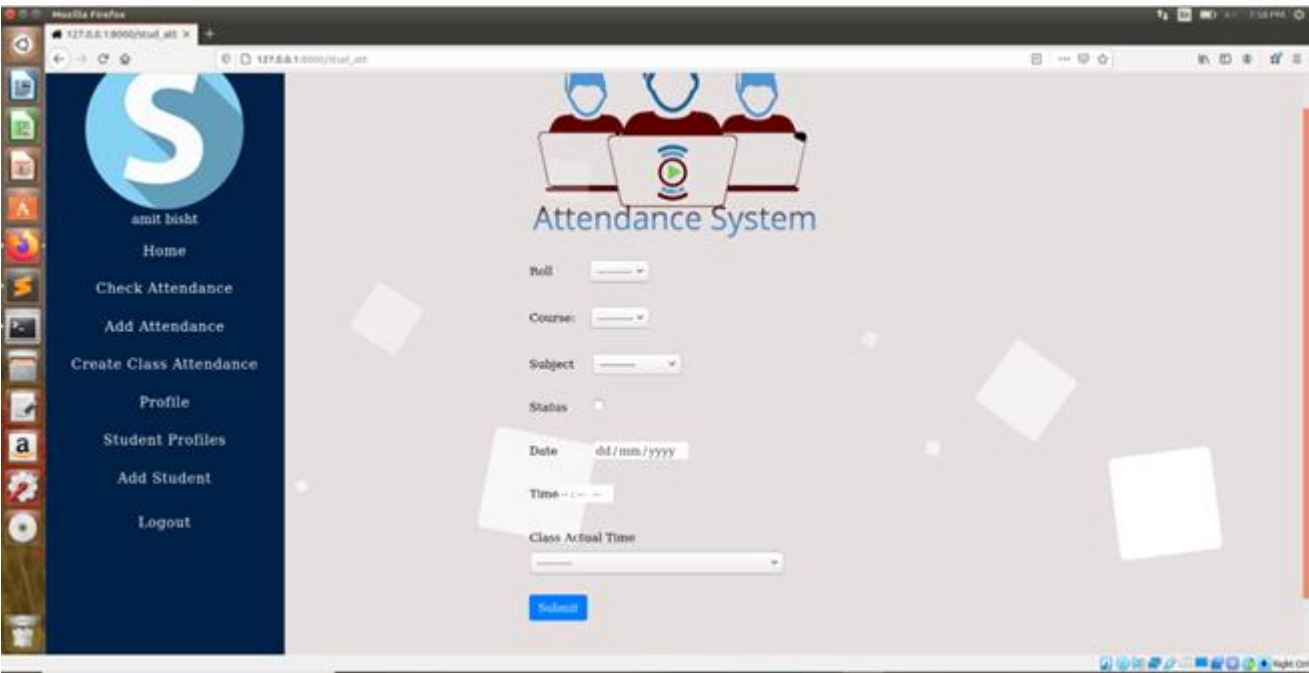
Teacher Check an Individual Student Attendance:



The screenshot shows a web browser displaying the Attendance System interface. On the left is a dark blue sidebar with a large white 'S' logo and the username 'amit bisht'. Below the logo are menu items: Home, Check Attendance, Add Attendance, Create Class Attendance, Profile, Student Profiles, Add Student, and Logout. The main content area displays a table with three columns: S.no, Date, and Status. The table contains seven rows of attendance data.

S.no	Date	Status
1	Sept. 18, 2020, 9 p.m.	False
2	April 30, 2020, noon	True
3	April 30, 2020, 5 a.m.	False
4	April 16, 2020, 6-30 a.m.	True
5	April 15, 2020, 8-30 a.m.	False
6	April 15, 2020, 6-30 a.m.	True
7	April 14, 2020, 6-30 a.m.	True

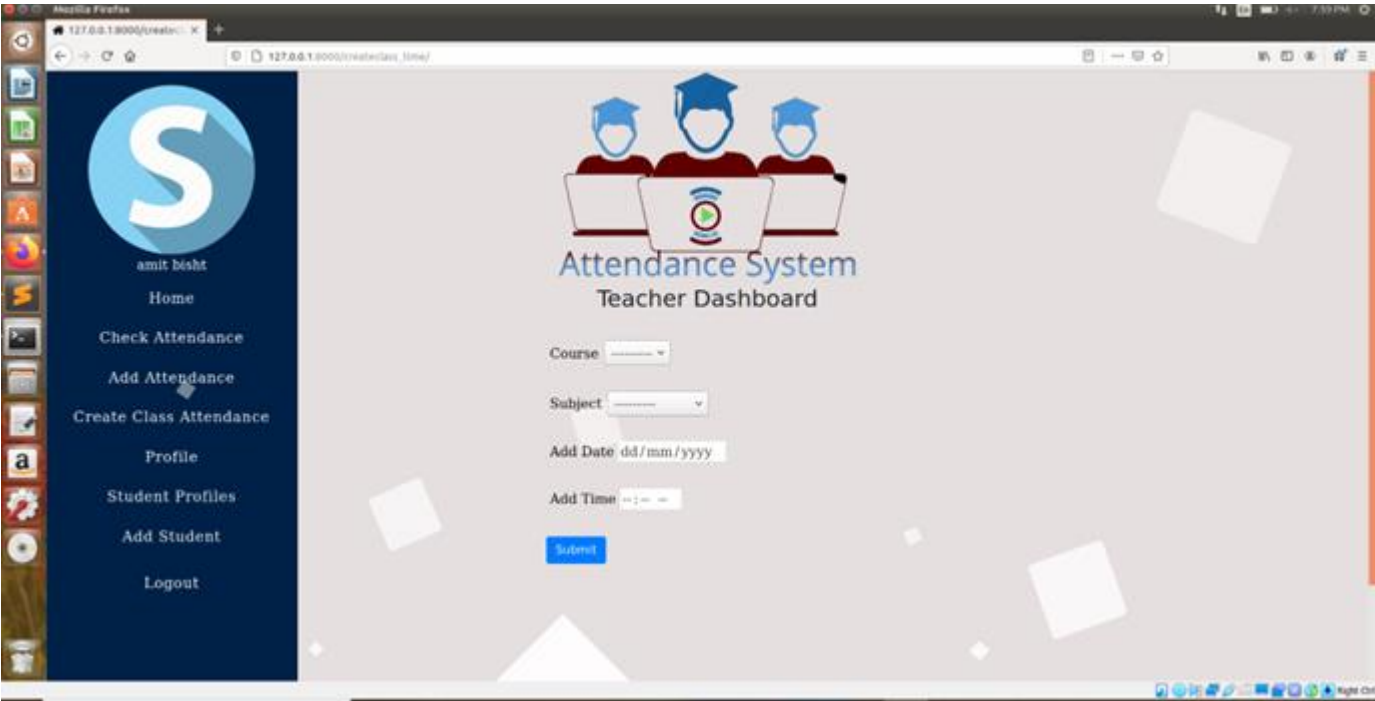
Teacher mark student teacher



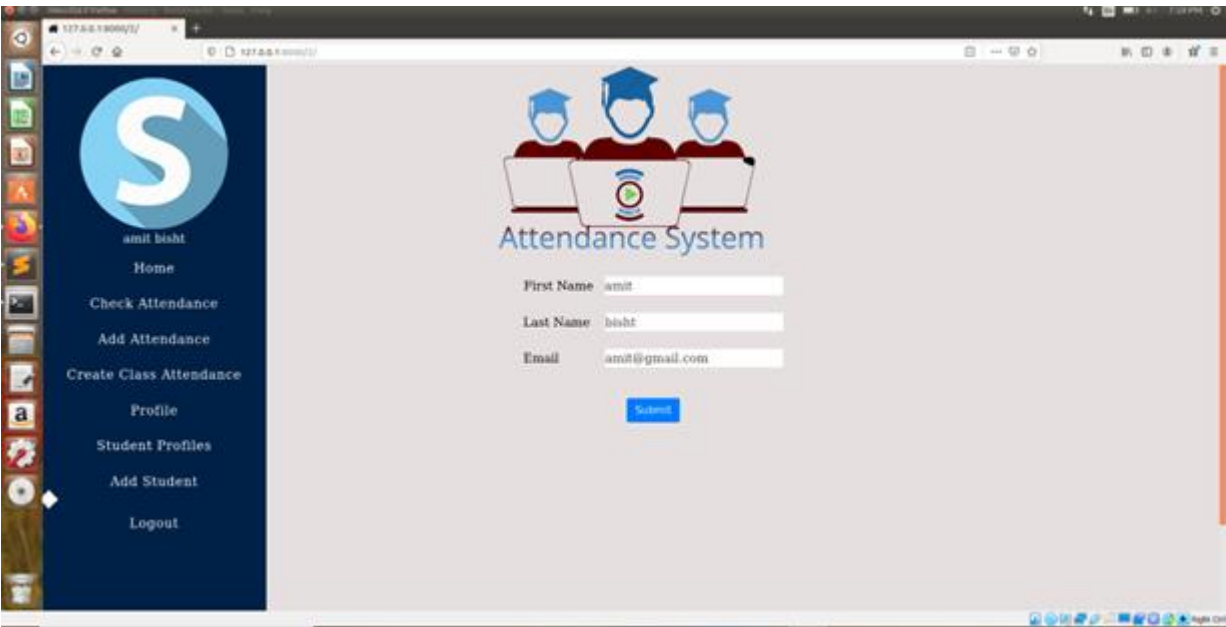
The screenshot shows the same web browser displaying the Attendance System interface. The main content area features a form titled 'Attendance System' with a header image of three students at laptops. The form includes dropdown menus for Roll, Course, and Subject, a radio button for Status, and input fields for Date (dd/mm/yyyy) and Time (hh:mm). There is also a 'Class Actual Time' dropdown menu and a 'Submit' button. The sidebar is identical to the previous screenshot.



Teacher Create Class Time:



Check Profile:



Teacher Check Student Profile:

amit bisht

Home

Check Attendance

Add Attendance

Create Class Attendance

Profile

Student Profiles

Add Student

Logout

S.no	Roll No	Student Name	Course Name	Email	Edit	Face Snapshot
1	7	Mahesh Bisht	BVOC	mahesh@gmail.com	<a href="#">Click Here</a>	<a href="#">Click here</a>
2	1	Rohannnnnn Singh	BCA	rohan@gmail.com	<a href="#">Click Here</a>	<a href="#">Click here</a>
3	4	Ashish Kumar	BVOC	ashish@gmail.com	<a href="#">Click Here</a>	<a href="#">Click here</a>
4	10	akash kumar	BCA	akash@gmail.com	<a href="#">Click Here</a>	<a href="#">Click here</a>
5	3	Manish Bisht	BVOC	manish@gmail.com	<a href="#">Click Here</a>	<a href="#">Click here</a>
6	11	Shubham Singh	BCA	shubham@gmail.com	<a href="#">Click Here</a>	<a href="#">Click here</a>

Teacher Add New Student:

amit bisht

Home

Check Attendance

Add Attendance

Create Class Attendance

Profile

Student Profiles

Add Student

Logout

Attendance System

Roll No

000

First Name

Last Name

Email

Course

Student

☒

Year

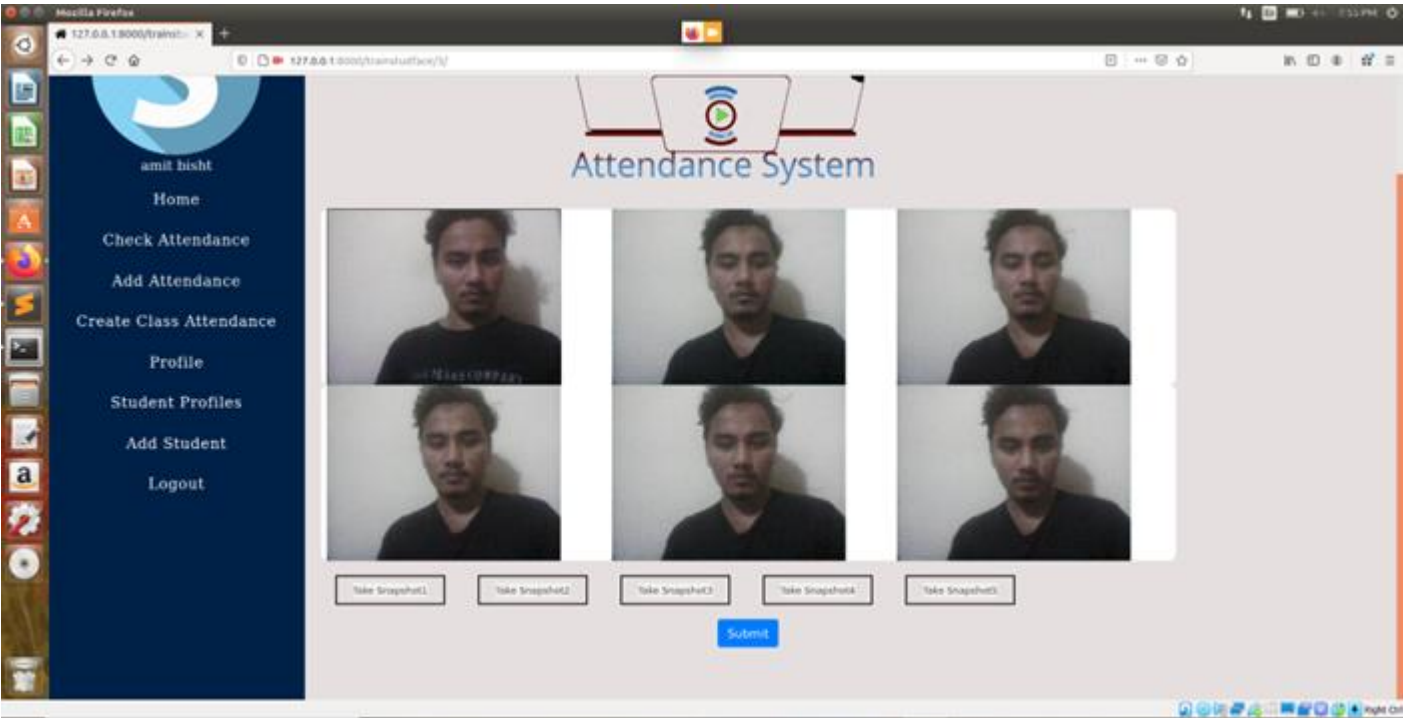
NA

Password

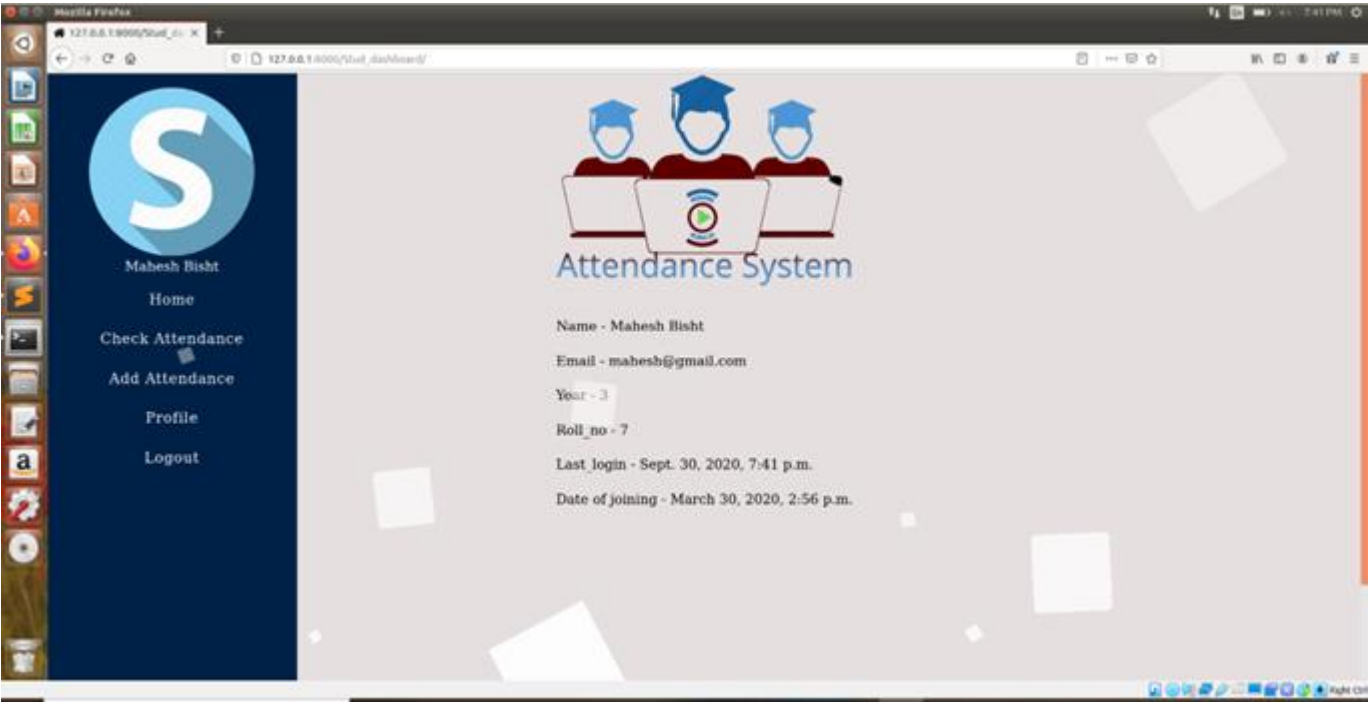
Password

Submit

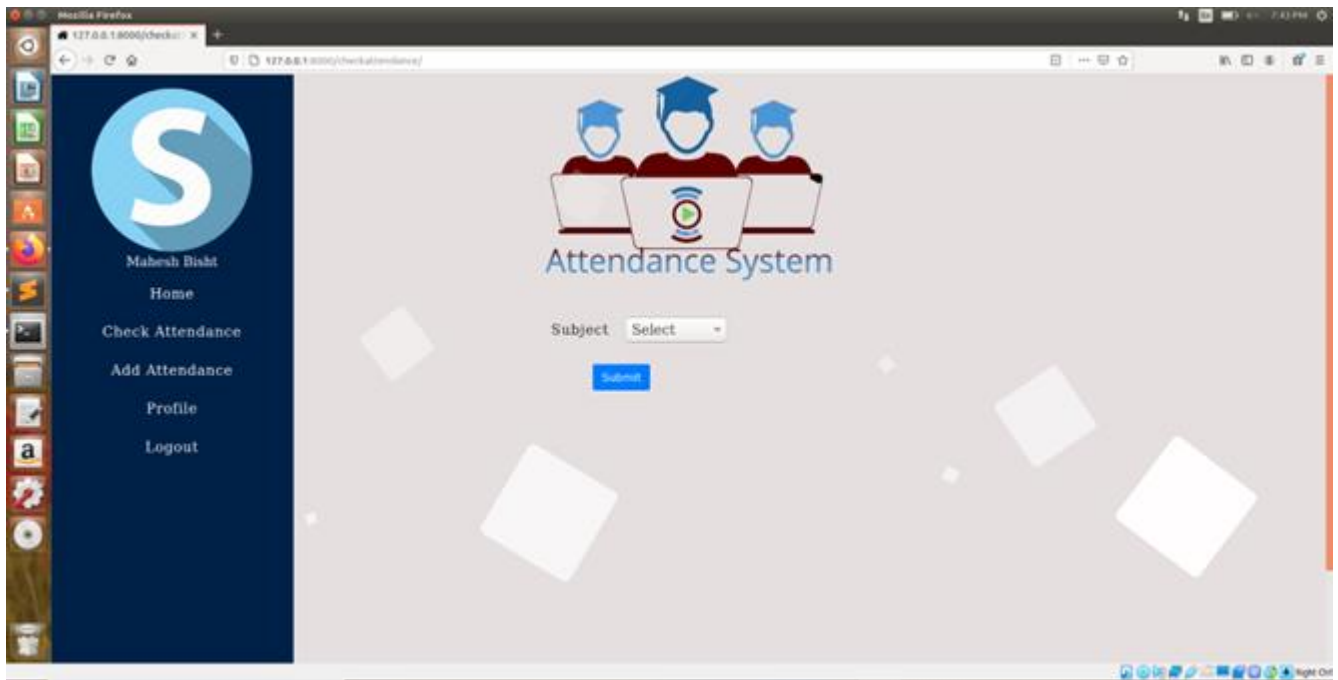
Teacher Training Student Face:



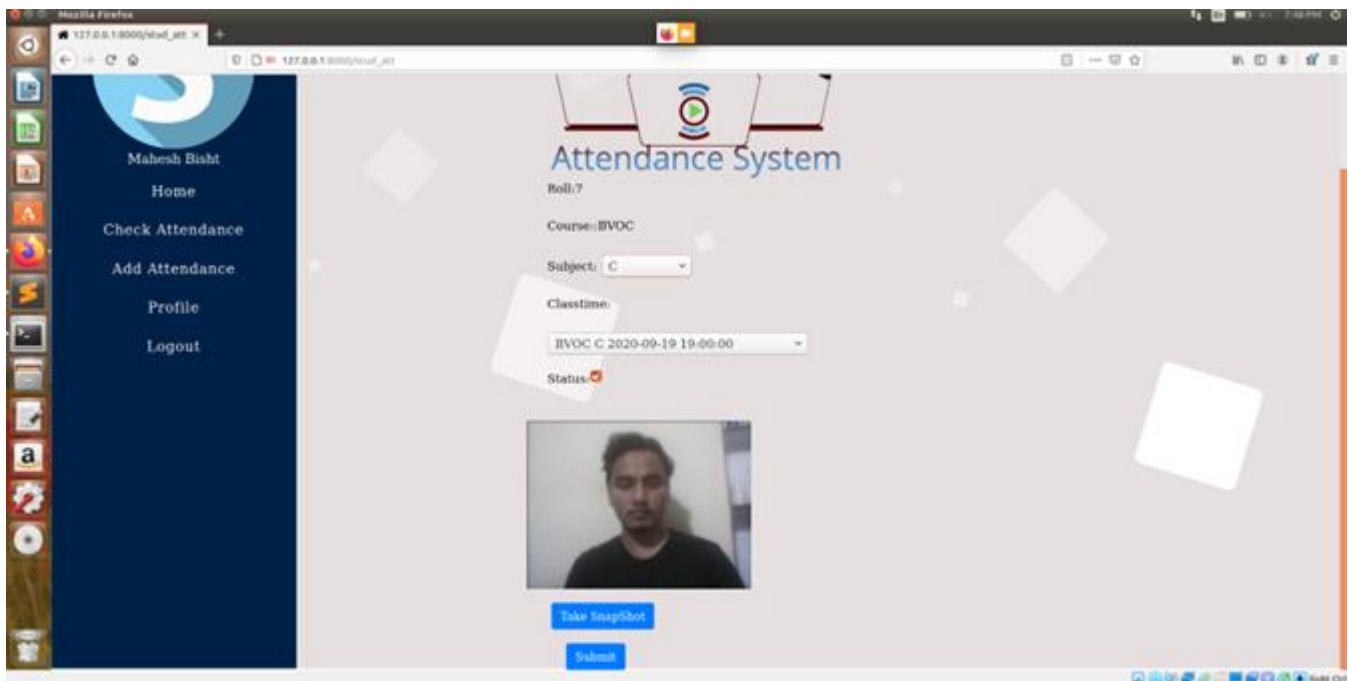
Student Dashboard



## Student Check Attendance



## Student Mark Attendance



Student Check Profile:



## **CHAPTER 9**

### **BIBLIOGRAPHY**

#### **HTML& CSS**

- W3School HTML/CSS Tutorials, References and Examples @ <http://www.w3schools.com/>.
- (W3School is not related to W3C).
- Matthew MacDonald, "Creating a Website - The Missing Manual", 3rd ed, 2011, O'Reilly.
- (A good introductory book on HTML/CSS.)
- Matthew MacDonald, "HTML 5 - The Missing Manual", 2nd ed, 2014, O'Reilly.
- David Sawyer McFarland, "CSS 3 - The Missing Manual", 3rd ed, 2013, O'Reilly.

#### **JAVASCRIPT &Jquery**

- W3School JavaScript Tutorials, References and Examples @ <http://www.w3schools.com>.
- jQuery Tutorial @ <https://learn.jquery.com>.
- David Sawyer McFarland, "JavaScript and jQuery - The missing manual", 3rd ed, 2014, O'Reilly.
- Jonathan Chaffer and Karl Swedbery, "Learning jQuery", 4th ed, 2013, Packt Publishing.

#### **DJANGO & PYTHON**

- Django tutorial @ <https://docs.djangoproject.com/en/1.8/intro/tutorial01/>
- Python Language Reference @ <https://docs.python.org/3/>

#### **MYSQL**

- MySQL Mother Site @ [www.mysql.com](http://www.mysql.com).
- MySQL 5.7 "Reference Manual" @ <http://dev.mysql.com/doc/>.
- MySQL 5.7 "SQL Statement Syntax" @ <http://dev.mysql.com/doc/refman/5.7/en/sql-syntax.html>.
- Paul DuBois, "MySQL Developer's Library", 4th ed, 2009 (5th ed is probably available).
- Russell Dyer, "MySQL in a Nutshell", 2nd ed, O'reilly, 2008.

