Princess Sumaya University for Technology
King Hussein School for Computing Sciences

Automatic Attendance System For PSUT Students – (AAS)

Using Facial Recognition Technology

Prepared By:
Amro Alshadfan-20150161
Amr Almuani-20150256

Supervised By:
Dr. Ammar Odeh

Project Submitted in partial fulfillment for the degree of Bachelor of Science in Computer Science

Summer 2019/2020

AUTOMATIC ATTENDANCE SYSTEM

Declaration of Originality

This document has been written entirely by the undersigned team members of the project. The source of every quoted text is cited and there is no ambiguity in where the quoted text begins and ends. The source of any illustration, image, or table that is not the work of the team members is also clearly cited. We are aware that using non-original text or material or paraphrasing or modifying it without proper citation is a violation of the university's regulations and is subject to legal actions.

Names and Signatures of team members:

Amro Almuani

Amro Alshadfan

AUTOMATIC ATTENDANCE SYSTEM

Acknowledgments

We are honored and proud to be students of Princess Sumaya University for Technology which is reputable for its high-quality performance and its special faculty.

We are indebted to Dr.Ammar Odeh our project advisor for his patience and perseverance tackling and following every single point of this project. He illuminated the scientific consequences for us so as to raise our achievement and the quality of our performance.

We would like to thank Dr.Ammar for his care, welfare and support he respectfully showed during his monitoring. Without his generous support and valuable guidance this work could not have been accomplished in such a successful way.

We extend and dedicate our success to our inspirational Dr Mustafa Alfayoome who inspired us with his brilliant idea which enabled us to launch our project.

We wish this unusual educational era PSUT and its staff all success all the time to keep nurturing more students to be ready for the future .

AUTOMATIC ATTENDANCE SYSTEM

Summary

In our fast-moving world and the huge development of the education system as a whole and the continuously increasing numbers of students both in universities and schools and the indispensable fact that attendance is a vital part of the education process.

Considering that students these days are more engaged and interested in their social life and other activities more than they are interested in class which certainly affects the much-required student-instructor relation and has a huge effect on the learning outcomes and as a result of these factors students marks will be critically low.

Moreover, with the increasing numbers of students in each course, the traditional method of taking attendance is wasting more and more time from the valuable lecture time that's why we came up with an idea that solves this problem

The Automatic Attendance System-AAS is a web-based system that uses a state-of-the-art technology of human identification using face recognition to automatically take student attendance through cameras.

Generally, the Automatic Attendance System will not only solve the problem of time-wasting in taking attendance but also the problem of internal transformation and provide an accurate evaluation of students' grades compared to their attendance and allow students to attended compensatory lectures in coordination with the instructor.

Our methodological approach is that the system has to be a web-based system for better synchronization and easier access backed with two python scripts for facial recognition and creating images Database for accurate operation.

AUTOMATIC ATTENDANCE SYSTEM

List of Abbreviations

List the abbreviations you have used in your project if there are any and what they stand for.

UML: Unified Modeling Language.

JSP: Java Server Pages.

DLL: Dynamic Link Library.

CMS: Content Management System.

AUTOMATIC ATTENDANCE SYSTEM

Table of Contents

Declaration of Originality 2
Acknowledgments 3
Summary 4
List of Abbreviations 5
Table of Contents 6
Table of Figures 9
Table of Tables 11
Chapter 1 Introduction 12
1.1 Overview 12
1.2 Problem Statement 12
1.2.1 Problem Description 12
1.2.2 Project Outcome 13
1.2.3 Target audience 13
1.3 Related Work 13
1.4 Document Outline 15
Chapter 2 Project Plan 16
2.1 Project Deliverables 16
2.2 Project Tasks 16
2.3 Risk Assessment 18
2.4 Cost Estimation 20
2.5 Project Management Tools 20
Chapter 3 Requirements Specification 21
3.1 Stakeholders 21
3.2 Platform Requirements 21
3.2.1: Hardware requirements: 21
3.2.1: Software requirements: 21
3.3Functional Requirements 22
3.3.1 User Registration and Login Requirements 24
FR1 Register new user 24
FR2 User sign in: 25
FR3 User Forgot Password: 26
3.3.2 Main System Functions After User Login 27

AUTOMATIC ATTENDANCE SYSTEM

FR4 Add student images (Admin) 27
FR5 Add courses and sections (Admin) 28
FR6 View assigned courses and sections (Instructor) 29
FR7 Take attendance (Instructor) 30
FR8 Update and add grades (Instructor) 31
FR9 Generate statistics (Instructor) 32
FR10 Send notifications (Instructor) 33
FR11 View registered courses attendance and grades and notifications (students) 34
3.4 Non-Functional Requirements 35
3.4.1 Usability 35
3.4.2 Security 35
3.4.3 Reliability 35
3.4.4 Performance 35
3.4.5 Availability 35
3.4.6 Scalability 35
4.1 Logical Model Design 36
4.1.1 Component Diagram 36
4.1.2 Class Diagram 37
4.1.3 Use case Diagram 39
4.1.3.1 Expanded Use Case Take Attendance 41
4.1.3.2 Expanded Use Case Add Student Images 42
4.1.4 Activity Diagrams 43
4.1.4.1 Add new user 43
4.1.4.2 Add student image to database 44
4.1.4.3 Add Courses and sections and assign instructors and add students to the section. 45
4.1.4.4 Take Attendance 46
4.1.4.5 Update grades 47
4.1.5 Sequence Diagrams 48
4.1.5.1 Login Sequence 48
4.1.5.1 Taking Attendance Sequence 49
4.1.6 Entity Relationship Diagram 50
4.2 Physical Model Design 51
4.2.1 Reports Design 51
4.2.1.1 Attendance Report 51
4.2.1.2 Grades Report 51
4.2.2 User Interface 52
4.2.2.1 Login User Interface 52
4.2.2.2 Forgot Password User Interface 53
4.2.2.3 User List (Admin) 53
4.2.2.4 Add New User(Admin) 54
4.2.2.4 Add User Image (Admin) 55
4.2.2.5 Courses Add(Admin) 55
4.2.2.6 Courses List (Admin) 56

AUTOMATIC ATTENDANCE SYSTEM

4.2.2.7 Course Management (Admin) 56
4.2.2.8 Add Section (Admin) 57
4.2.2.9 Section Management (Admin) 57
4.2.2.10 Instructor Dashboard 58
4.2.2.11 Take Attendance (Instructor) 59
4.2.2.13 Update Grades (Instructor) 60
4.2.2.14 Send Notification (Instructor) 60
4.2.2.15 Student Dashboard 61
4.2.3: Database Design: 62
4.2.3.1: Users Table: 62
4.2.3.2: Administrator Table: 62
4.2.3.3: Instructor Table: 62
4.2.3.4: Student Table: 63
4.2.3.5: Course Table: 63
4.2.3.5: Section Table: 63
Chapter 5 Implementation 64
5.1 Programming Languages ..... **Error! Bookmark not defined.**
5.1.1 Front-End ..... **Error! Bookmark not defined.**
5.1.2 Back-End ..... **Error! Bookmark not defined.**
5.2 Tools And API ..... **Error! Bookmark not defined.**
5.3 General Implementation Overview ..... **Error! Bookmark not defined.**
5.4 Main Algorithms and code convention ..... **Error! Bookmark not defined.**
5.4.1 User Creation and editing ..... **Error! Bookmark not defined.**
5.4.2 Course creation 70
5.4.3 Section creation 70
5.4.4 Adding student to the image dataset 71
5.4.5 Taking attendance of a section 72
5.5 Requirements Coverage 75
Chapter 6 Testing 77
6.1 Testing Approach 77
6.2 Testing Tools 77
6.3 Testing Sample 78
6.4 Tested Features 80
Chapter 7 Conclusions and Future Work 82
Appendix A Users' Manual 83
Appendix B Document Changes 95
Appendix C Code Documentation 96
References 97

AUTOMATIC ATTENDANCE SYSTEM

Table of Figures

Figure 1:Task Table Figure 16
Figure 2:Gantt Chart 17
Figure 3:Component Diagram 36
Figure 4:Class diagram 38
Figure 5:Usecase Diagram 40
Figure 6:New user activity diagram 43
Figure 7:Add student image activity diagram 44
Figure 8:Adding courses activity diagram 45
Figure 9:Take attendance activity diagram 46
Figure 10:Update grades activity diagram 47
Figure 11:Login sequence diagram 48
Figure 12:Taking Attendance Sequence diagram 49
Figure 13:Entity Relationship Diagram 50
Figure 14:Attendance Report Design 51
Figure 15:Grades Report Design 51
Figure 16:Login Page 52
Figure 17:Password Reset Page 53
Figure 18:User List Page 53
Figure 19:Add New User Page 54
Figure 20:Add User Image Page 55
Figure 21:Adding Courses 55
Figure 22:Course List Page 56
Figure 23:Course Management Page 56
Figure 24:Add Section Page 57
Figure 25:Section Management Page 57
Figure 26:Instructor Dashboard Page 58
Figure 27:Taking Attendance Page 59
Figure 28:Update Grades Page 60
Figure 29:Send Notification Page 60
Figure 30:Student Dashboard 61
Figure 31:Users Table 62
Figure 32: Administrator Table 62
Figure 33:Instructor Table 62
Figure 34:Student Table 63
Figure 35:Course Table 63
Figure 36:Section Table 63
Figure 37: Dataset Creation Algorithm 67
Figure 38:Face Recognition Algorithm 68
Figure 39:Dataset image example 72
Figure 40:Face recognition example 75

AUTOMATIC ATTENDANCE SYSTEM

Figure 41:User manual – login page 83
Figure 42: User manual – login page2 84
Figure 43: User manual – Password reset 84
Figure 44: User manual – Password reset 2 85
Figure 45 :User manual – Dashboard 85
Figure 46: User manual – logout 86
Figure 47: User manual – Password change 1 86
Figure 48: User manual – Password change 2 87
Figure 49 : User manual – Add new user 1 87
Figure 50 :User manual – Add new user 2 88
Figure 51 :User manual – Add student face 1 88
Figure 52 :User manual – Add student face 2 89
Figure 53 : User manual – Add course 1 89
Figure 54 : User manual – Add course 2 89
Figure 55 : User manual – Add course 3 90
Figure 56 : User manual – Add section 1 90
Figure 57 : User manual – Add section 2 90
Figure 58 : User manual – Add section 3 91
Figure 59 : User manual – Take attendance 1 91
Figure 60 : User manual – Take attendance 2 92
Figure 61 : User manual - View attendance 1 92
Figure 62 : User manual – View attendance 2 93
Figure 63 : User manual – Adding grades 1 93
Figure 64 : User manual – Adding grades 2 93
Figure 65 : User manual – Adding grades 3 94
Figure 66 :User manual – Adding grades 4 94
Figure 67: Code Documentation 1 96
Figure 68 : Code Documentation 2 96

AUTOMATIC ATTENDANCE SYSTEM

Table of Tables

Table 1:Document Outline 15
Table 2:Tasks Table Description 17
Table 3:Risk Assessment Table 18
Table 4:Cost Approximation Table 20
Table 5:Stakeholders 21
Table 6:Functional Requirements 22
Table 7: Register new user Register new user 24
Table 8: User sign in 25
Table 9: User Forgot Password 26
Table 10: Add student images 27
Table 11: Add courses and 28
Table 12: Add courses and sections 29
Table 13: Take attendance 30
Table 14: Update and add grades 31
Table 15: Generate statistics 32
Table 16: Send notifications 33
Table 17: View registered courses attendance and grades and notifications 34
Table 18: Expanded Use Case Take Attendance 41
Table 19:Expanded Use Case Add Student Images 42
Table 20:Tools And API ..... **Error! Bookmark not defined.**
Table 21:Requirement Coverage 75
Table 22:Test Sample report 78
Table 23:Requirement Coverage 80
Table 24:Document changes 95

AUTOMATIC ATTENDANCE SYSTEM

Chapter 1
Introduction

1.1 Overview

Automatic Attendance System is a web-based system designed to help instructors to take attendance in a fast and accurate way and obviate the use of traditional attendance taking the method of taking attendance by calling out each student by name which forms a big issue especially in non-English speaking countries where names are hard to spell and pronounce correctly.

The importance of this project lies in helping instructors take advantage of the cameras located in each classroom to take attendance automatically and accurately using equipment that already exists in the classroom which saves the precious lecture time and to help instructors to evaluate students' performance based on the vital relationship with their grades in quizzes and exams.

We chose facial recognition over other effective identification methods in light of the current COVID-19 situation which makes any physical interaction requiring methods such as fingerprints forms a high risk of infection on public health and safety while facial recognition does not require any physical interaction with the identification equipment what so ever eliminating any risk on public health.

Also, it's worth mentioning that our system can be implemented to serve not only educational institutes but also business companies that wish to monitor their employee's attendance and movement within the office and it can be easily implemented in preexisting systems.

Also, we chose it to develop our knowledge and experience in the field of biometrics and automated security systems.

Our web system aims to reach every instructor, school leadership, and student in Jordan in its first stage.

1.2 Problem Statement

1.2.1 Problem Description

Attendance to classes is a vital part of the academic process without attending lectures the student cannot acquire the skills and knowledge of the course nor build a healthy relationship with his instructor.

Usually, academic classes start by the lecturer taking attendance of students and in most cases this process is a manual process of calling the names of all students registered, not to mention that students are less interested in coming to class these days and most of the classes

AUTOMATIC ATTENDANCE SYSTEM

have a huge number of students registered so taking students attendance will be a hectic time-consuming process.

Moreover, attendance is entered manually to the system therefore it is are error prone. Our new system will maintain safety and security for students to keep parents and guardians well informed and prevents students' truancy.

1.2.2 Project Outcome

The final output of the system will be a web-based system that follows the regulations and rules of the web-based system with proper security measures, and we hope that it will solve problems like student attendance and student evaluation and internal exchange on time.

1.2.3 Target audience

This web-system is dedicated to every academic institute, teacher, student, and school leadership in The Hashemite Kingdom of Jordan and around the world.

1.3 Related Work

The field of using biometrics has expanded rapidly over the past decade and many projects where made in this field using many identification and tracking methods Like RFID, Fingerprints and Facial recognition and IoTs.

As proposed by [1], students Fingerprint is registered upon enrollment and a Fingerprint scanning device is fixed in each lecture room then when a student enters he/she scans his/her finger through the device fixed in each lecture room and at the end of each semester reports are extracted to a C# system that operates locally and it analysis the reports to determine if the student is eligible to take an exam or not.

Reference [2] proposed a system that takes attendance using Radio frequency identification (RFID) readers that are distributed within the work space and every employee is issued a badge or a card that is used to be swiped on the RFID reader which saves the information with time of checking in or out and then the data is downloaded manually by the system administrator.

as in reference [2], reference [3] proposed another system that also uses RFID by issuing each student a magnetic card and track these cards through RFID readers fixed at each lecture room and when a student enters the lecture room his card is automatically scanned and sent to a C# interface that sends an attendance report to the instructor after the lecture via email.

In reference [4] authors proposed an attendance system that takes attendance using iris scanning, the main idea is that the student iris is registered through a scanner then stored in a local database and iris scanners are fixed at lecture room door when a student enters he/she uses the scanner to scan iris which compares the scanned iris with the database then generates a report and sends it to the managing PC through the wireless network.

AUTOMATIC ATTENDANCE SYSTEM

Reference [5] Proposed a system that uses QR code to register attendance where the instructor logs in and generates a QR code then the instructor displays the QR code to students to scan using their phones and generates a report while having a condition that the students must be connected to the wireless network of the lecture room.

Reference [6] Proposed a face recognition system where students face is scanned and the face parameters are detected and stored in a local database and to take attendance the student stand Infront of a pc connected with a camera that takes a picture and converts it to parameters and compares it to the local database

Reference [7] Proposed a system that uses facial recognition which requires the administration to add students images to a local database and requires the instructor to record a video which contains all students who attend the class then input it to a local system that scans the video and detects the faces of students then generates a local attendance report.

Our work:

Due to the current COVID-19 pandemic, we decided to choose an identification method that does not require any physical interaction with devices or computers to maintain public health and safety also we searched for a method that achieves integrity and accurately does the required task with minimal risk of manipulation and deception so we chose facial recognition because it provides the most accurate and honest results and saves more time compared to other technologies like RFID, Fingerprints, Iris Detection, and QR codes we previously discussed.

Our system will be a web-based system that operates online and provides live high-speed face recognition while only using equipment that is already available in every institute and without the need of any special hardware unlike many of the previous proposals we discussed earlier and we hope to achieve this while maintaining ease of use and a friendly user interface to even support users with no technological experience, we also implement a unique feature which compares student attendance with there grade and fully functional management and communication system.

AUTOMATIC ATTENDANCE SYSTEM

1.4 Document Outline

Table 1:Document Outline

Chapter Title	Description
1. Chapter 1: Introduction	Provides a brief overview of the project's idea and the problem that it solves also gives an assumption of the targeted audience and previous similar work.
2. Chapter 2: Project Plan	Provides a detailed description of the outcomes of the project and the tasks that will be fulfilled and the distribution of these tasks between team members and a risk assessment and cost estimation and tools.
3. Chapter 3: Requirements Specifications	Describes the functional and non-functional requirements also includes the needed software and hardware specifications with the declaration of the stakeholders.
4. Chapter 4: System Design	Provides a detailed analysis for the system components both high- and low-level design using UML diagrams.
5. Chapter 5: Implementation	Provides a detailed description of the code and the overall programming phase of the project.
6. Chapter 6: Testing	Describes the testing techniques and testing tools used in this project.
7. Chapter 7: Conclusion and Future Plans	Provides a summary of the fulfilled tasks and project outcome and discusses plans of development.

AUTOMATIC ATTENDANCE SYSTEM

Chapter 2
Project Plan

2.1 Project Deliverables

Our final product of this project will be a responsive web-based system that uses python scripts that run in the back-end to create a Database of students' images and identify students and a front end that uses HTML, CSS, PHP, and JAVASCRIPT, also a full documentation and code samples.

2.2 Project Tasks

TASK NAME	START DATE	DAY OF MONTH*	END DATE	TEAM MEMBER	PERCENT COMPLETE
Analysis					
General Idea and Problem Statement	6/25	3	6/28	Both	100%
Idea Applicability Search	6/29	2	7/1	Both	100%
Finding Requirments	7/1	1	7/3	Both	100%
Requirment Modeling	7/3	3	7/6	Both	100%
Searching For Data Collection Methods	7/6	6	7/7	Both	100%
Choosing Data Collection Method	7/7	7	7/9	Both	100%
Development Strategy	7/9	9	7/10	Both	100%
Design					
User Interface Design	7/11	11	7/12	Both	100%
DataBase Design	7/13	13	7/15	Both	100%
Implementation strategies	7/16	16	7/20	Both	100%
System Design	7/20	20	8/3	Both	100%
Implementation (Next Semester)					
Coding					0%
Testing					0%
Final Documentation					0%
Publish Calculator					0%

Figure 1:Task Table Figure

AUTOMATIC ATTENDANCE SYSTEM

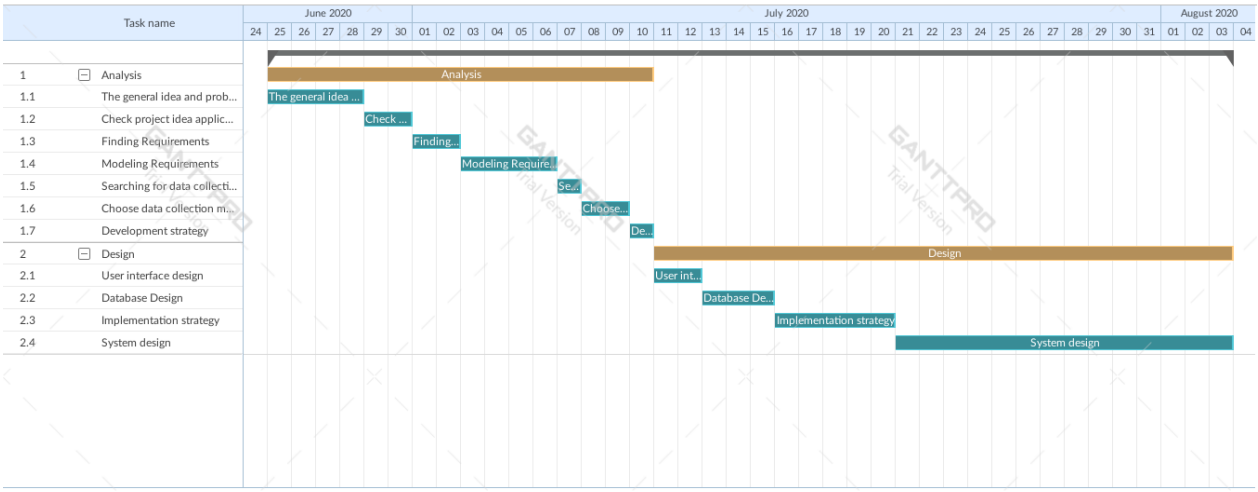

Figure 2:Gantt Chart

Table 2:Tasks Table Description

Task Number	Task Name	Description	Time (day)	Dependencies	Task Responsibility
2.2.1 Phase 1: Analysis					
T1	The general idea and problem statement	To provide a detailed description of the general idea of the project and the problem we hope to solve	4	None	A.Alshadfan A.Almuani
T2	Check project idea applicability	To search and discuss if our project can be implemented from a technical point of view	3	T1	A.Alshadfan A.Almuani
T3	Finding Requirements	To find both functional and non-functional requirements and other requirements of our system	3	T2	A.Alshadfan A.Almuani
T4	Modeling Requirements	To draw diagrams of the requirements we identified	4	T3	A.Alshadfan A.Almuani

AUTOMATIC AUDIANCE

T5	Searching for data collection methods	To search for appropriate methods to get student information	2	T4	A.Alshadfan A.Almuani
T6	Choose the data collection method	To choose a method from the methods that we found in the previous task	3	T5	A.Alshadfan A.Almuani
T7	Development strategy	After we identify the requirements, we need to put a development strategy which includes deciding which software and hardware is going to be needed	2	T6, T3	A.Alshadfan A.Almuani
2.2.2 Phase 2: Design					
T8	User interface design	To design the user interface and the front-end experience of the system	2	T7	A.Alshadfan A.Almuani
T9	Database Design	To design the database schema	3	T8	A.Alshadfan A.Almuani
T10	Implementation strategy	To decide how we are going to implement the system and prevent any diversion of the requirements we identify	5	T9, T3	A.Alshadfan A.Almuani
T11	System design	To design the diagrams such as class diagram, use case, activity diagrams and others	15	T10, T3	A.Alshadfan A.Almuani
Phase 3: Implementation (Next Semester)					

AUTOMATIC ATTENDANCE SYSTEM

2.3 Risk Assessment

Table 3:Risk Assessment Table

Task Number	Task Name	Risk Number	Risk	Probability	Impact
Phase 1: Analysis					

T1	The general idea and problem statement	R1	Incomplete explanation of the project's idea and problem	Low	Less understanding of our project
T2	Check solution applicability	R2	If our way of solving project problem is inapplicable in the real world or not accepted by stakeholders	Moderate	Project cancelation
T3	Finding Requirements	R3	To fail to identify all vital requirements of our system	Low	Missing an important feature
Phase 2: Design					
T8	User interface design	R4	Failing to fulfill user needs	Moderate	Fewer institutes use our platform
T9	Database Design	R5	Error in the Database schema	Moderate	The system not functioning properly
T10	Implementation strategy	R6	Failing to choose the right programing language that suits our system need	Low	Failing to achieve our goals of the project
T11	System design	R7	Failing to provide a correct design	Low	System not functioning properly
Phase 3: Implementation (Next Semester)					

AUTOMATIC ATTENDANCE SYSTEM

2.4 Cost Estimation

An Approximation of the costs of this System:

Table 4:Cost Approximation Table

Web Costs		
Service	**Provider**	**Cost/Year**
Server Hosting	GoDaddy	120$
Domain Name	Namecheap	11$
Hardware Costs		
Service	**Provider**	**Cost/Year**
Camera		140$
Other Costs		
Service	**Provider**	**Cost/Year**
Maintenance	Team Members	100$

2.5 Project Management Tools

During the time of Covid-19 pandemic we needed tools that allow us to work together remotely and synchronize our work to achieve this we will use:

- Google Drive(web)[8]: Storage of our latest work such as charts, diagrams or document
- Lucidchart(web)[9] and StarUML (v3.2.2)[10]: Used to design Charts and UML diagrams and Database schema easily.
- GanttPro(web)[11]: used for Gantt chart creation
- Brackets(web)[12]: will be used to synchronize our code implementation

AUTOMATIC ATTENDANCE SYSTEM

Chapter 3
Requirements Specification

3.1 Stakeholders

Table 5:Stakeholders

Stakeholder	Description
Admin	• Adds students and instructors and courses to the system • Adds students and assigns to courses • Add students' images to the system database • System maintenance
Instructor	• One of the main targeted audience • Manages courses he teaches • Takes attendance of students • Enter marks for each course he teaches • Communicates with students
Student	• One of the main targeted audience • View attendance and marks of each course registered

3.2 Platform Requirements

3.2.1: Hardware requirements:

Since the system is web-based and will have a python back-end the webserver must support Django services and has a big capacity to accommodate the huge amount of data that needs to be processed and stored and must be a high-speed server to handle the load of users.

3.2.1: Software requirements:

Users will only need a computer with a camera connected and a fast internet connection

AUTOMATIC ATTENDANCE SYSTEM

3.3Functional Requirements

Table 6:Functional Requirements

Requirement ID	Requirement Name	Requirement Detail	Priority
FR1	Register new user	To register a new user to the system and add it to the database	Essential
FR2	Sign in	To allow the user to log in to his/her account login the system	Essential
FR3	Forgot password	Recovering user password in case he/she forgets and setting up a new password for his/her account	Essential
FR4	Add student images	Adding students' images to the database to prepare for face recognition	Essential
FR5	Add courses and sections	Adding new courses and sections to the system and add students to the section and assign an instructor to each section	Essential
FR6	View assigned courses	For instructors to view the courses they are teaching, and the students registered in each section	Essential
FR7	Take attendance	To take attendance of students for each lecture using face recognition	Essential
FR8	Update and add grades	Managing grades for each course	Essential
FR9	Generate statistics	Generating a report for each course section that shows	Recommended

AUTOMATIC ATTENDANCE SYSTEM

		the instructor how attendance affects grades	
FR10	Send Notifications	The ability to send notifications to the students	Recommended
FR11	View registered courses attendance and grades and notifications	View registered courses attendance and grades and notifications	Essential

AUTOMATIC ATTENDANCE SYSTEM

3.3.1 User Registration and Login Requirements

FR1 Register new user

Table 7: Register new user Register new user

Function	Add new users to the system and store it in the Database.
Description	Admin enters new user required information and roll and if the information does not exist in Database the new user information will be stored in the Database.
Input	user type (student or instructor, admin), id, name, email, major, phone number, year of admission, password.
Outputs	If a field is missing or the user already exists in Database an error message is shown and if the operation is successful a message is shown, and an email is sent to the newly registered user email.
Constrains	ID must not exist in the Database and the password must include upper- and lower-case characters and numbers and symbols of 8 characters.
Process	Admin will have the field to choose between 3 user types (admin, instructor, student) and a field to enter ID according to what type of user he chooses. If admin user type is chosen full name and email and password and phone number are also required along with admin ID. If instructor user type is chosen full name and department and email and password and phone number are required along with instructor ID If student user type is chosen full name and major and email and password and phone number are required along with student ID. If the ID does not exist in the database and the operation is successful a message will appear to the admin that the operation was successful, and an email will be sent to the user saying that he is registered
Priority	Essential

AUTOMATIC ATTENDANCE SYSTEM

FR2 User sign in:

Table 8: User sign in

Function	User login to the system using his/her account
Description	On the login portal user enters id and password if they are entered correctly, they will be redirected to Dashboard.
Input	user-input ID and password
Outputs	If the information entered by the user is not correct an error message will appear, and the user will be prompted to request that he/she forgot password function (F3).
Constrains	ID and password must match the information stored in the Database
Process	When the user enters the login page he/she is required to enter ID and password if the information entered is correct the user is redirected to the appropriate dashboard corresponding to the user's type if the information entered is not correct an error message is shown and he/she will be asked if he/she wants to request the forgot password function (F3)
Priority	Essential

AUTOMATIC ATTENDANCE SYSTEM

FR3 User Forgot Password:

Table 9: User Forgot Password

Function	Rest user password if he/she forgets it.
Description	if the user forgets his/her password he/she enters ID and corresponding email to reset the password.
Input	ID and email
Outputs	If the information entered by the user is correct and matching information in the Database an email will be sent to his registered email containing a password reset link.
Constrains	ID and email must match the records in the Database.
Process	When the user chooses to click forget password he is redirected to a page in which he/she will enter ID and email if the entered information matches the records in the database an email will be sent to the user's registered email containing a link to reset the password when the link is clicked user is asked to enter a new password after the new password is entered its recorded in the Database and the user is redirected again to the login page.
Priority	Essential

AUTOMATIC ATTENDANCE SYSTEM

3.3.2 Main System Functions After User Login

FR4 Add student images (Admin):

Table 10: Add student images

Function	Add student image for facial recognition
Description	Admin enters student ID and starts a script that takes a picture of the student and stores it in the Database
Input	Student ID
Outputs	ID must be found in the Database records
Constrains	ID and email must match the records in the Database.
Process	Admin will have the field to enter student ID and Full name and a button to start the image storing script if the ID entered are not corresponding to a student in the Database an error message is displayed if its found the image storing script is started and images of the student are taken automatically and stored in the Database to be used in face recognition and taking attendance by instructors. If the process is successful a message will be displayed.
Priority	Essential

AUTOMATIC ATTENDANCE SYSTEM

FR5 Add courses and sections (Admin):

Table 11: Add courses and

Function	Adding courses and sections of lectures and add students to them
Description	Admin creates courses and sections for each course taught in the institution
Input	Course ID, Course Name, Section ID, Section Time, Section instructor ID
Outputs	If student ID doesn't exist an error message is shown and if an operation is successful, the proper message is shown
Constrains	Course ID must not match any other record in the Database.
Process	Admin will have the field to enter student ID and Full name and a button to start the image storing script if the ID entered are not correspond Admin will have the field to enter Course ID and name and if the course ID already exists an error message is displayed and if the operation is successful admin is redirected to a page to create sections for every course each section has an ID and time and instructor ID if the section ID is duplicated or instructor doesn't exist error message is shown after a section is created successfully admin is redirected to a page in which he/she has a field to enter student IDs that are registered in this course and if IDs does not exist in the database an error message is shown and if the operation is successful an appropriate message is shown and admin is redirected to create more sections. adding to a student in the Database an error message is displayed if its found the image storing script is started and images of the student are taken automatically and stored in the Database to be used in face recognition and taking attendance by instructors. If the process is successful a message will be displayed.
Priority	Essential

AUTOMATIC ATTENDANCE SYSTEM

FR6 View assigned courses and sections (Instructor):

Table 12: Add courses and sections

Function	View courses and sections that the instructor teaches.
Description	The ability to view information about every course and section the instructor teaches, and view students registered in each one.
Input	None
Outputs	Courses and sections assigned to the instructor and students enrolled in each one.
Constrains	None
Process	When an instructor is logged in, he/she can view the courses and sections assigned to him/her and view the details of the students enrolled and in each one. Also, an instructor can start the auto attendance function (F4) and can start managing grades function for each section (F5) and send notifications using the function (F7) and start statistics generating function (F6)
Priority	Essential

AUTOMATIC ATTENDANCE SYSTEM

FR7 Take attendance (Instructor):

Table 13: Take attendance

Function	Taking attendance automatically
Description	The ability to view information about every course and section the instructor teaches, and view students registered in each one.
Input	Course ID, section ID
Outputs	Attendance report.
Constrains	Course ID and section ID must be valid and exist in the Database
Process	When an instructor is logged in, and courses are displayed by function (F6)when a section is clicked students information is displayed and an option to start the face recognition script when that option is clicked instructor has to enter Course ID and section ID, if both are valid script automatically, starts identifying students faces and comparing them with images added by the admin in function (F4) if a match is found the student is checked if he is registered in the course he will be added to the report as attended and then the script checks students who are not identified but registered in the class and adds them as absent to the attendance report after that a CSV report is generated for instructor view and automatically sent to be imported in the Database. If a student passes 4 absences an alert notification t is automatically sent by the function (3.3.2.7) also if a student passes the 7 absences a notification to contact the instructor is sent to the student.
Priority	Essential

AUTOMATIC ATTENDANCE SYSTEM

FR8 Update and add grades (Instructor):

Table 14: Update and add grades

Function	Updating and add students' grades.
Description	The ability to add and manage grades for each student registered in each section.
Input	Grades
Outputs	None
Constrains	must be from a 0 to 100 scale
Process	When an instructor is logged in, and courses are displayed by function (F6)when a section is clicked students information is displayed and an option to add and update grades of the section when that option is clicked registered student list is displayed with 4 fields for each student for the instructor to enter 1st,2nd,3rd and final grade, and the instructor can download grades report
Priority	Essential

AUTOMATIC ATTENDANCE SYSTEM

FR9 Generate statistics (Instructor):

Table 15: Generate statistics

Function	Generating statistics for each course
Description	The ability to generate a statistic showing how attendance affected student grades
Input	None
Outputs	Statistics file
Constrains	None
Process	When an instructor is logged in, and courses are displayed by function (F6)when a section is clicked students information is displayed and an option to add and update grades of the section when that option is clicked registered student list is displayed with 4 fields for each student for the instructor to enter 1st,2nd,3rd and final grade, and the instructor can download grades report
Priority	Recommended

AUTOMATIC ATTENDANCE SYSTEM

FR10 Send notifications (Instructor):

Table 16: Send notifications

Function	Sending notifications to students
Description	The ability to send notifications to students by the instructor
Input	Message
Outputs	None
Constrains	None
Process	When an instructor is logged in, and courses are displayed by function (F6) when a section is clicked students' information is displayed and an option to send a notification to students registered in the section
Priority	Recommended

AUTOMATIC ATTENDANCE SYSTEM

FR11 View registered courses attendance and grades and notifications (students)

Table 17: View registered courses attendance and grades and notifications

Function	To view the courses a student is enrolled in and attendance and marks and notifications
Description	The ability to view which courses a student is registered in and the information related.
Input	None
Outputs	None
Constrains	None
Process	When a student is logged in, courses that he/she is displayed and when clicked on he/she is redirected to a page containing the number of absence times and course grades and notifications sent by the instructor.
Priority	Essential

AUTOMATIC ATTENDANCE SYSTEM

3.4 Non-Functional Requirements

3.4.1 Usability:

The Automatic Attendance System web user interface must be easy to understand and work with even for users with limited technological experience which forces the UI to be self-explaining and responsive.

3.4.2 Security:

Each user's information must be stored in an encrypted format especially user passwords and images and to protect from malicious and spam user registration the only users authorized to add new users are system admins.

3.4.3 Reliability:

The system must be able to handle user requests on time and be crash-free as much as possible will be achieved by extensive deep testing to mitigate all vulnerabilities and code errors.

3.4.4 Performance:

Since the main goal of our system is to save time the system could be able to run smoothly and accurately and must accomplish tasks fast so students' pictures are resized before being uploaded to the Database and the Design and backend scripts are focused on being light and fast loading.

3.4.5 Availability:

The system must be available as needed except the maintenance period and must be able to handle all request amounts promptly and handle a lot of logins especially during lecture time.

3.4.6 Scalability

The system could handle the increasing number of users and large amounts of data and have a responsive design that allows it to be accessed from all devices that can be extended to emulate to other systems.

3.5 Domain Requirements

There are no limits to the functionality of this system as it can be customized according to the institute's needs and requirements.

AUTOMATIC ATTENDANCE SYSTEM

Chapter 4

System Design

4.1 Logical Model Design

4.1.1 Component Diagram

Component Diagrams describe the main components of the system and how they connect.

The first component is the system user that interacts with the 2nd component which is the backend through an interface and the backend gets the data from the database which is the 3rd component of the system

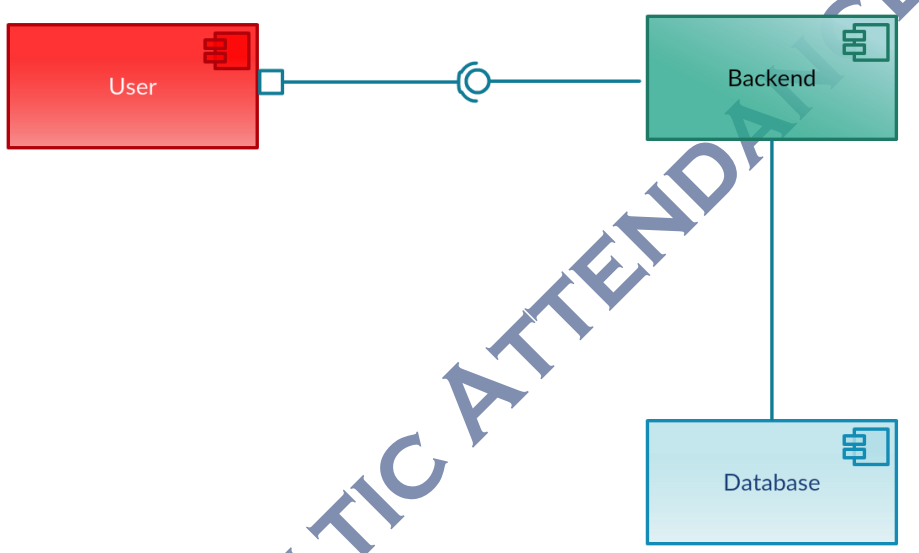

Figure 3:Component Diagram

AUTOMATIC ATTENDANCE SYSTEM

4.1.2 Class Diagram

Class diagrams are used to describe the structure of the system and the roles of each class and help in translating models into code.

In our system, we have three types of users: Administrators, instructors, and Students.

Administrators: manage the system they create users for both instructors and students and add students' images to the facial recognition system and create courses and sections and assign instructors and add students to them.

Instructors: view courses and take attendance and update grades and send notifications to students.

students can only view their registered courses and attendance and grades for each course and view notifications sent by the instructor.

AUTOMATIC ATTENDANCE SYSTEM

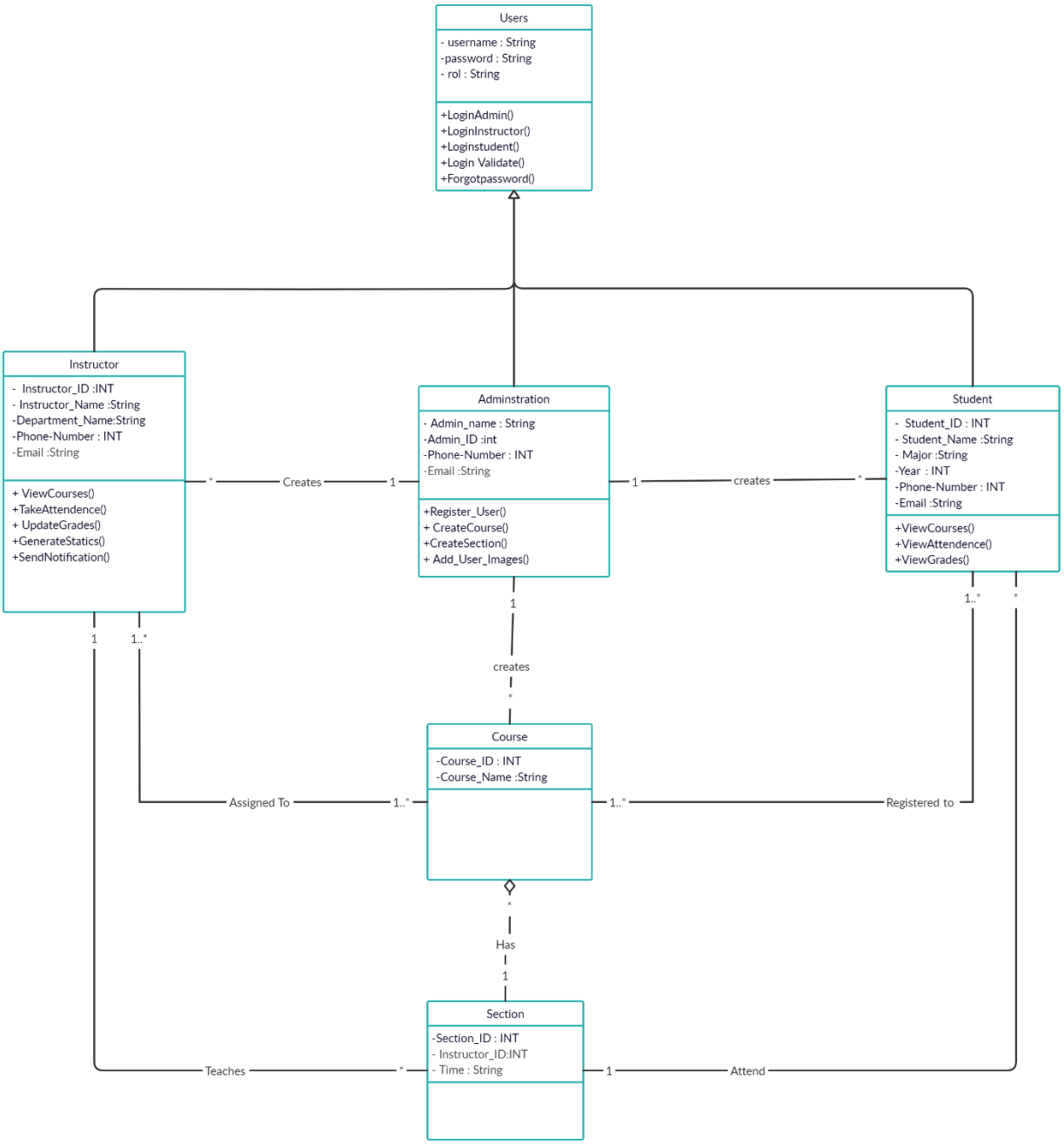

Figure 4:Class diagram

4.1.3 Use case Diagram

Use case diagram describes the basic interaction between the system and the user in different scenarios.

The Administrator

- Admin Log in to the system which also includes authentication of the username and password
- Creates a new user which includes checking if the user already exists
- Creates a new course and sections which includes checking if the course and section already exists and extends the ability to assign an instructor and add students to the section
- Add student images to the system which includes scanning the face of the student through the camera

The Instructor

- Instructor Log in to the system which also includes authentication of the username and password
- Views the course sections that instructor teaches during the semester
- Take attendance of a specific course section which includes scanning student's faces and can generate an attendance report if requested by the instructor and this includes that the instructor edit that reports before approving it
- Enters grades for each section he teaches
- Sends notifications to all students or a specific student in the section
- For each course, admin can generate a comparison between student grades and their attendance

The Student

- Views the courses and section times that he/she is registered in
- View student attendance for each course registered
- View grades each course registered
- View notifications received from the instructor

AUTOMATIC ATTENDANCE SYSTEM

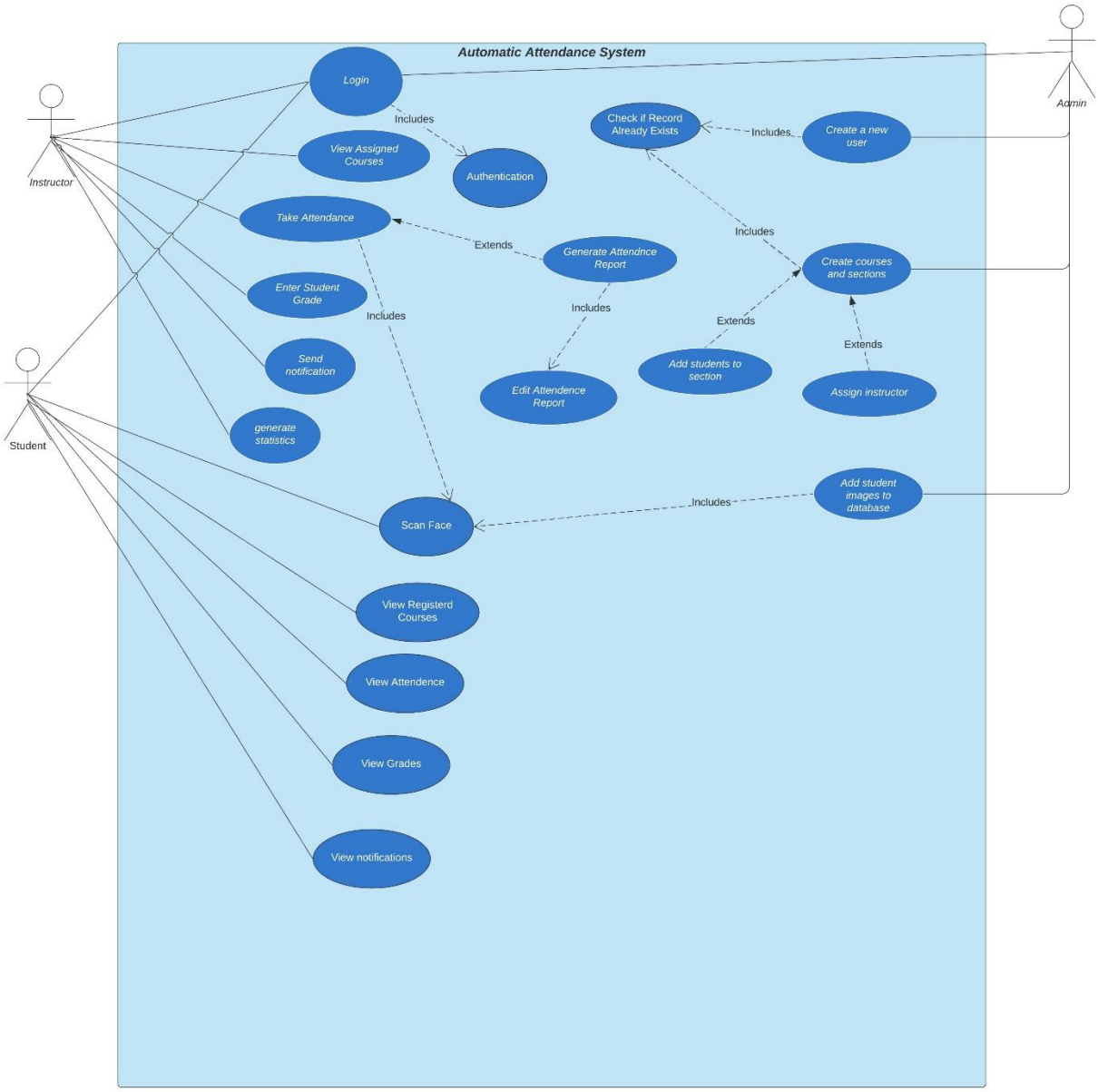

Figure 5:Usecase Diagram

AUTO

4.1.3.1 Expanded Use Case Take Attendance

Table 18: Expanded Use Case Take Attendance

Use Case:	Take Attendance
Actors:	Instructor(initiator), Student
Purpose:	To take attendance of students in each section
End(success):	generates an accurate report of attendance and stores it in the database.
End(failure):	If the report is not accurate then it's edited by the instructor before storing it in the database If the script fails to execute an error message is displayed and the report is not generated
Description:	After the instructor logs in he/she chooses a section and with a click, the attendance taking script is started and the student's faces are scanned through the camera and compared with the list of students registered in the course then added to an attendance report and this report is displayed to the instructor for his approval. If approved the report is stored in the database directly and if not the instructor is asked to edit the report manually and it's stored in the database after editing is done.
Type:	Essential
Reference:	Functions: FR2, FR7

Typical situation:

Actor Actions	System Response
1. The instructor enters the valid Login information	2. System displays dashboard
3. The instructor selects a course section from the list of courses in the dashboard	4. The system retrieves selected section information and registered students and displays it to the user.
5. Instructor clicks the take attendance button	6. The system starts the attendance script successfully and scans students faces The system generates a report of attendance according to the faces identified and displays it to the instructor
7. Instructor approves the report	8. System stores the report in the database

AUTOMATIC ATTENDANCE SYSTEM

Alternative situation:

Line 1: The instructor enters the wrong login information. The system displays an error message and asks the instructor to enter the information again.

Line 3: The system fails to execute the script or fails to generate a report. The system displays an error message

Line 4: The instructor doesn't approve of the report. The system asks the instructor to edit the report then stores it.

4.1.3.2 Expanded Use Case Add Student Images

Table 19:Expanded Use Case Add Student Images

Use Case:	Add Student Images
Actors:	Administrator(initiator), Student
Purpose:	To add student image to the database for face recognition
End(success):	scans student's face and adds it to the database
End(failure):	Student's face is not added to the database
Description:	After the administrator logs in, he/she chooses a student from the students' list and clicks a button that starts a script that scans the student face through the camera and saves it in the database
Type:	Essential
Reference:	Functions: FR2, FR4

Typical situation:

Actor Actions	System Response
1. Administrator enters the valid Login information	2. System displays dashboard
3. Administrator selects a student from the student list	4. The system retrieves selected student information and displays it to the Administrator
5. Administrator clicks the add image button	6. The system starts the face-scanning script and scans the student's face and adds it to the database then displays a message that operation was successful

Alternative situation:

Line 1: The administrator enters the wrong login information. The system displays the error message and asks the Administrator to enter the information again.

Line 3: The system fails to execute the script or fails to access the camera or fails to detect a student's face. The system displays the error message and asks the Administrator to retry.

AUTOMATIC ATTENDANCE SYSTEM

4.1.4 Activity Diagrams

4.1.4.1 Add new user

when a new user is to be added to the system an administrator must log in using his username and password and then the username and password is authenticated then the administrator enters the new user's information and that information is displayed to the administrator for confirmation then the entered information is checked if its already in the database if not the new user is recorded in the database.

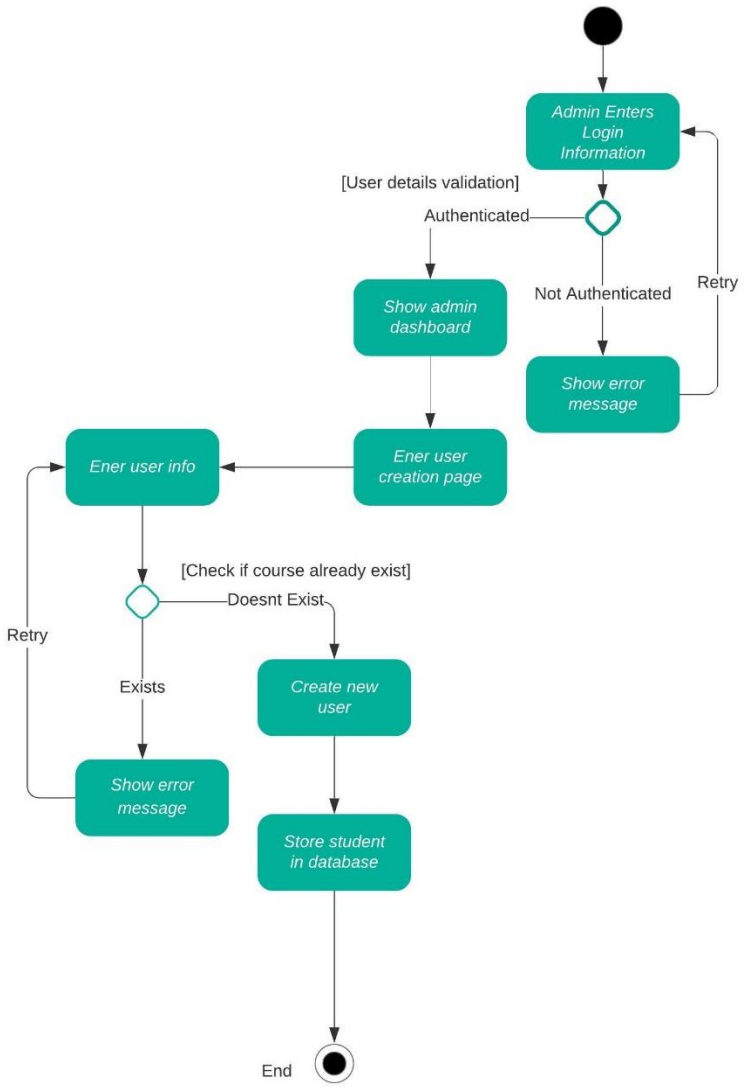

Figure 6:New user activity diagram

NCE SYSTEM

4.1.4.2 Add student image to database

For the face recognition to work a picture of the student must exist in the database so the system an administrator must log in using his username and password and then the username and password is authenticated then the administrator enters the student information and the system checks if the information entered is in the database then the administrator starts the backend image taking a script that automatically takes an image of the student and stores it in the database.

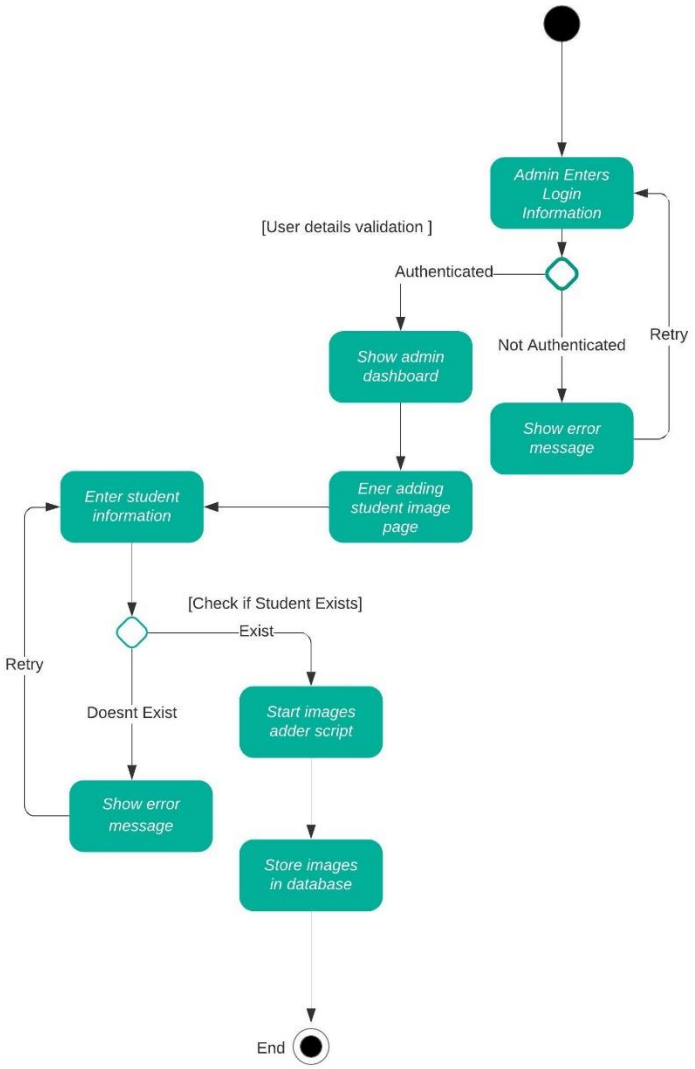

Figure 7:Add student image activity diagram

4.1.4.3 Add Courses and sections and assign instructors and add students to the section.

the administrator must log in using his username and password and then the username and password is authenticated then the administrator enters the course and section information and the system checks if the information entered already exists in database then the administrator has to enter the id of instructor and students id then the information is checked and stored in the database.

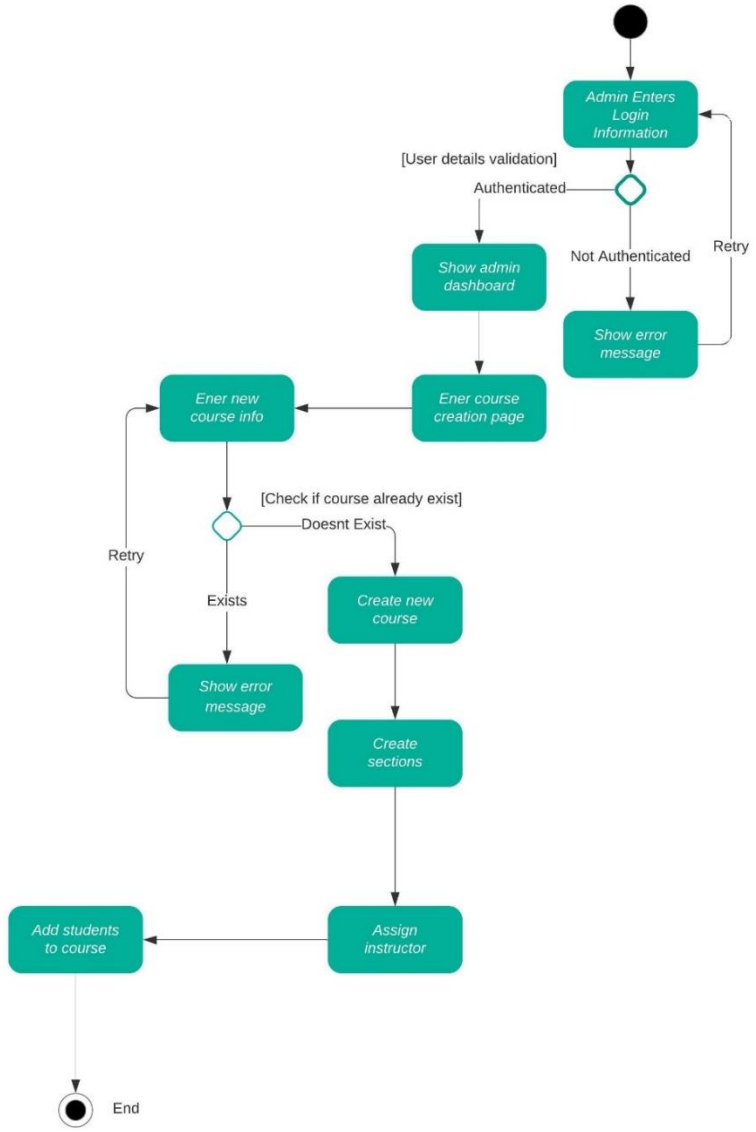

Figure 8:Adding courses activity diagram

4.1.4.4 Take Attendance

To take attendance instructor must log in using his username and password and then the username and password is authenticated then the instructor chooses the course and section then a script in the backend takes the student attendance and generates a report the instructor reviews the report then it's stored in the database and the student can log in and view it.

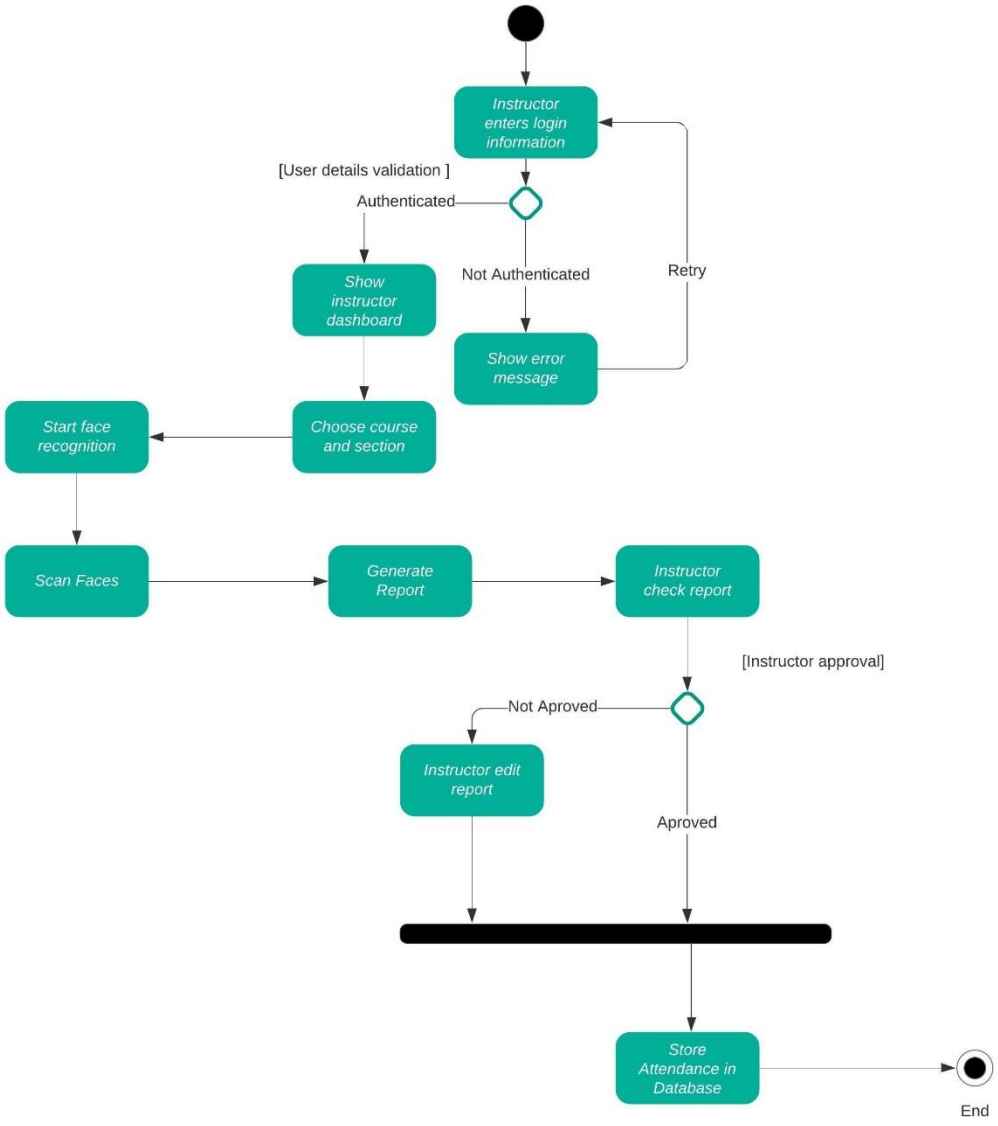

Figure 9:Take attendance activity diagram

4.1.4.5 Update grades

To update grades instructor must log in using his username and password and then the username and password is authenticated then the instructor chooses the course and section then adds grades to students registered in each section then a final report is shown to the instructor if the instructor approves the grades are stored in the database.

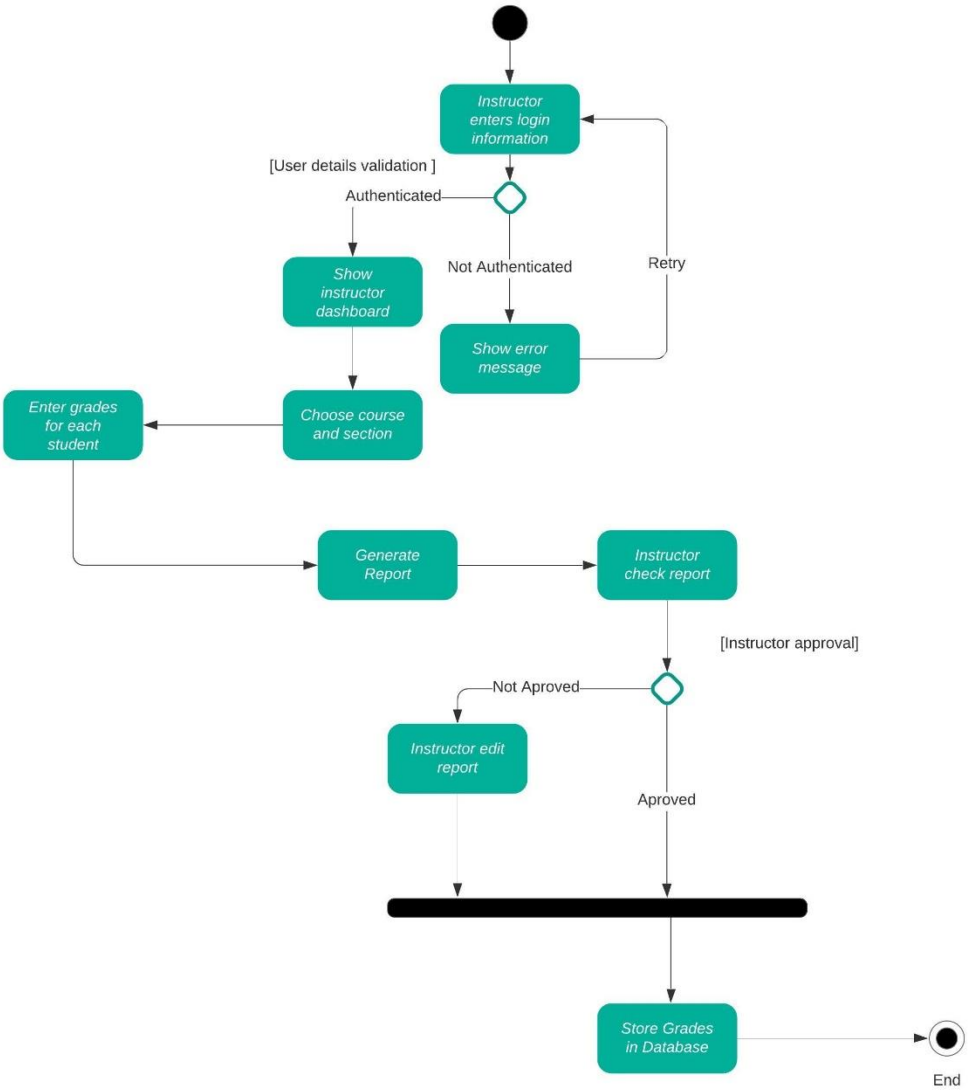

Figure 10:Update grades activity diagram

4.1.5 Sequence Diagrams

4.1.5.1 Login Sequence

Describes the sequence of operations when signing into the system

Where the student enters the info and the backend authenticates it from the database and if it is ok the user is redirected to the appropriate dashboard and if not, the user is shown an error message and requested to enter the login information again.

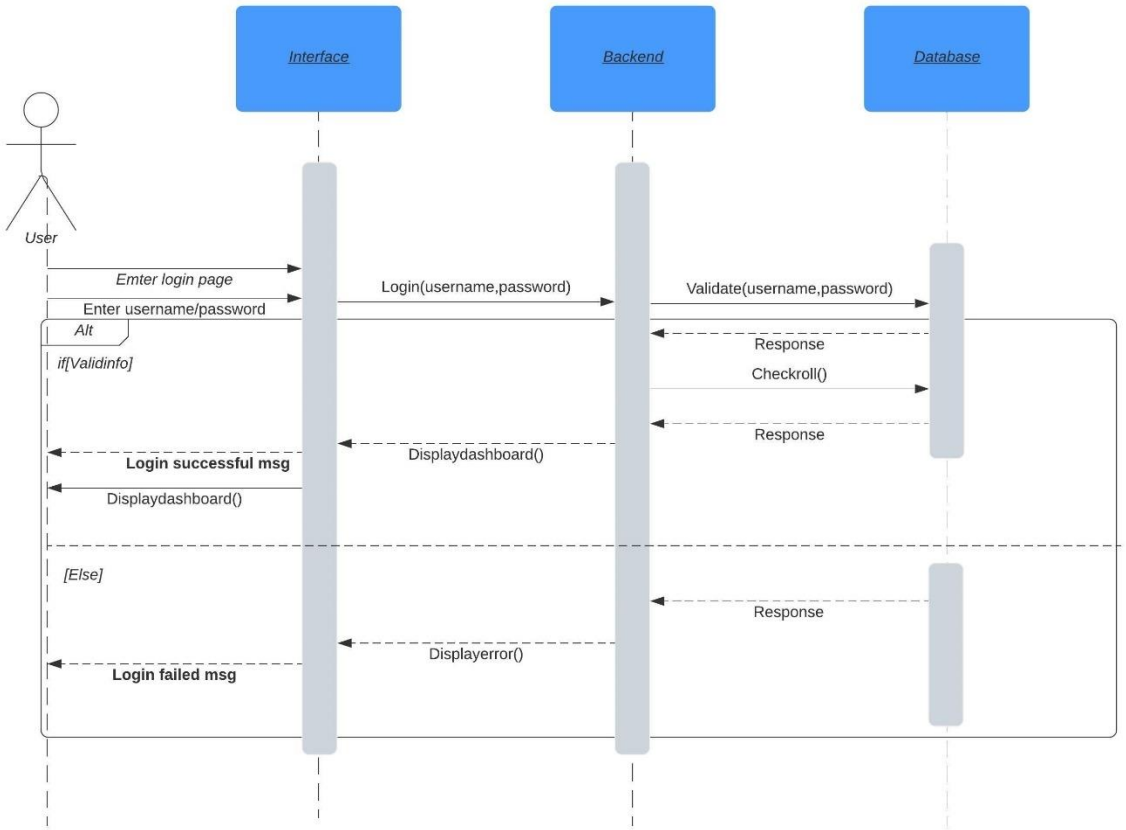

Figure 11:Login sequence diagram

4.1.5.1 Taking Attendance Sequence

Describes the sequence of operations when the instructor takes attendance

Where the instructor selects the section and starts the backend script which scans the faces of students and puts their information in a report which is shown later to the instructor for approval or edit before storing it to the database.

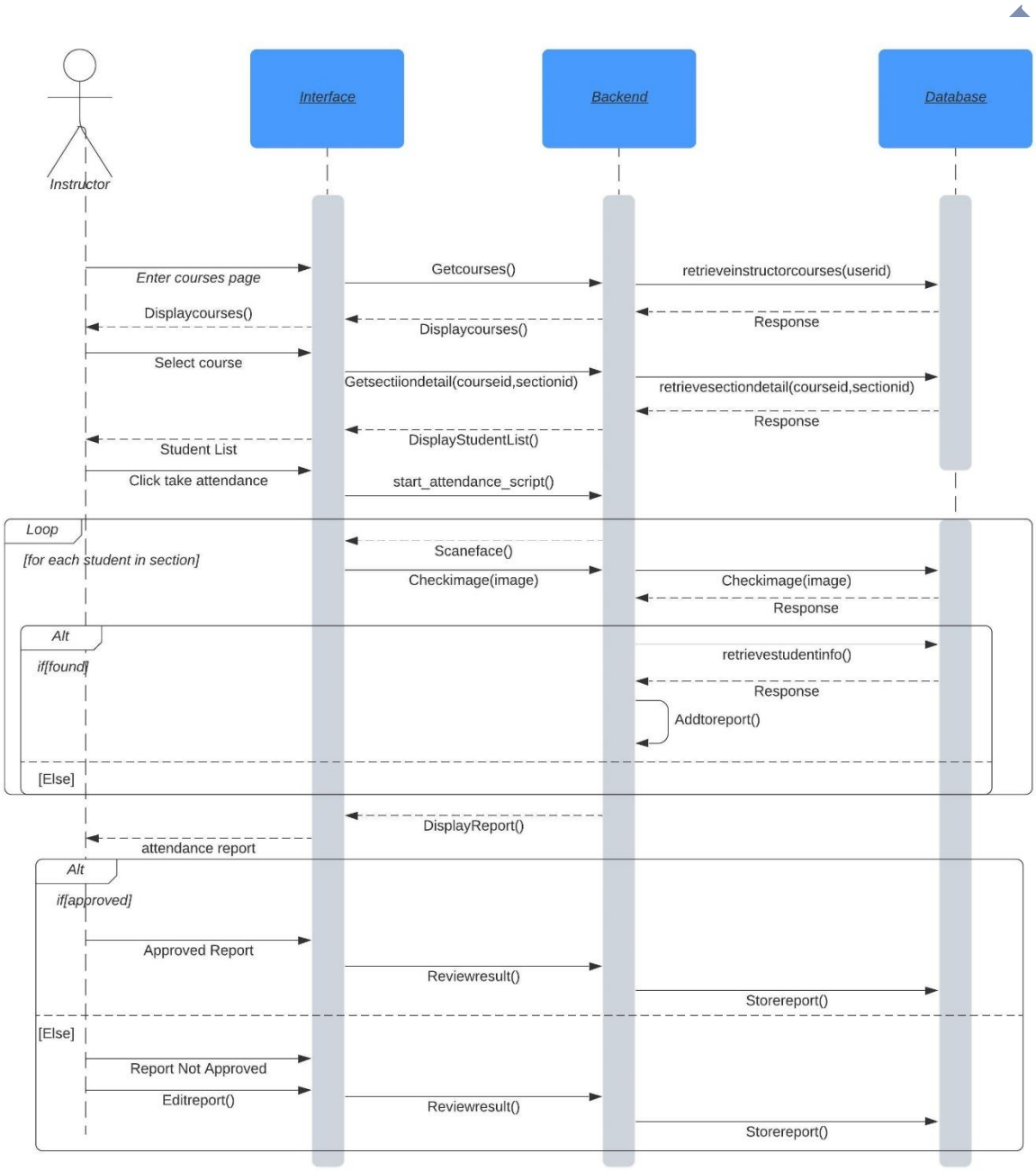

Figure 12:Taking Attendance Sequence diagram

4.1.6 Entity Relationship Diagram

An entity-relationship diagram (ERD) shows the relationships of entity sets stored in a database

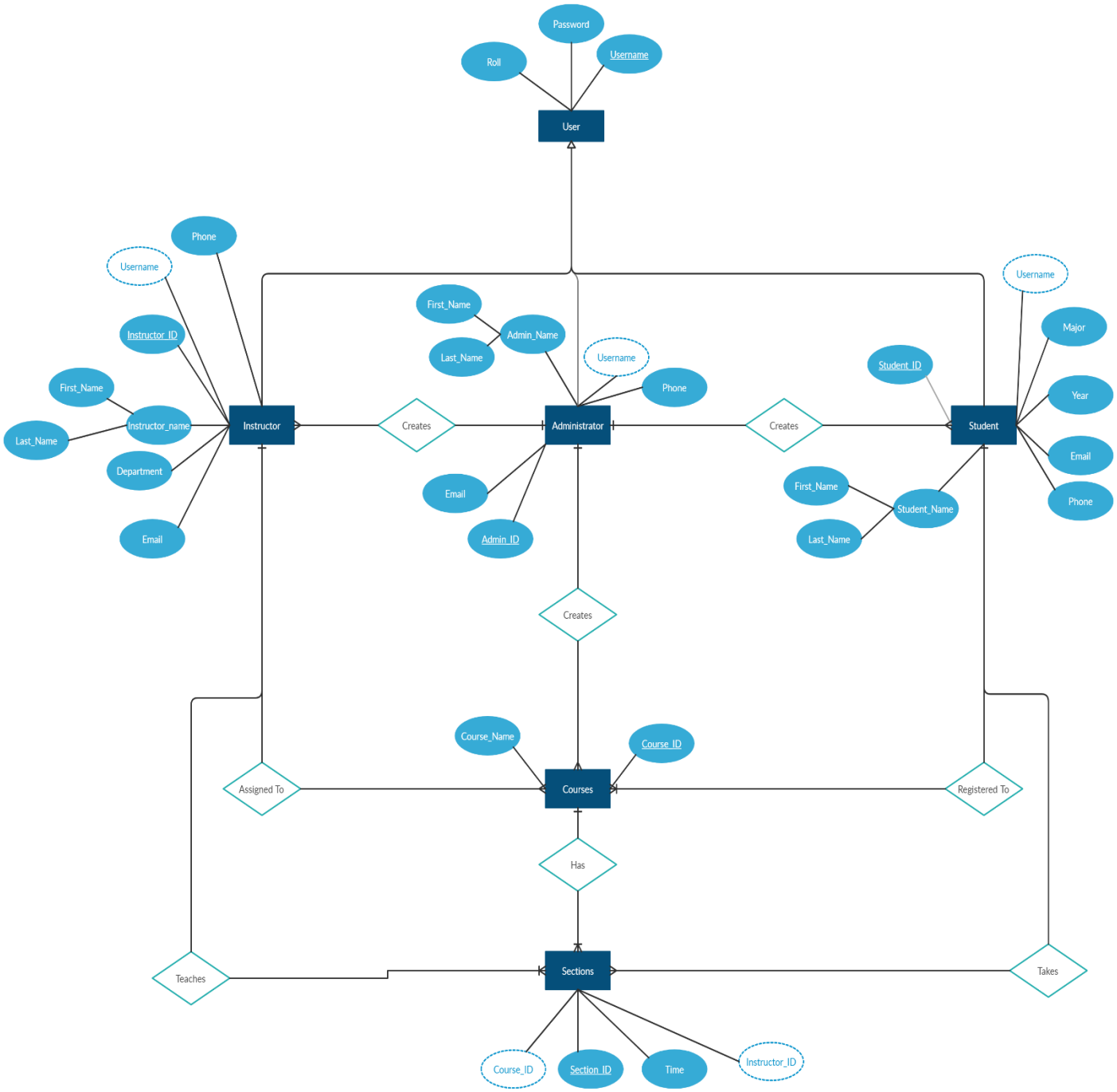

Figure 13:Entity Relationship Diagram

4.2 Physical Model Design

4.2.1 Reports Design

4.2.1.1 Attendance Report

Course ID			
Section ID			
Instructor ID			
Date			
	Full Name		
Student ID	First name	Last name	Attendance status

Figure 14:Attendance Report Design

4.2.1.2 Grades Report

Course ID						
Section ID						
Instructor ID						
	Full Name					
Student ID	First name	Last name	1st grade	2nd grade	3rd grade	Total grade

Figure 15:Grades Report Design

AUTOMATIC ATTENDANCE SYSTEM

4.2.2 User Interface

4.2.2.1 Login User Interface

The landing page will be the login page in which the user of the system enters his assigned username and password then if the information is valid the user will be redirected to a dashboard that suits his role (admin or instructor or student).

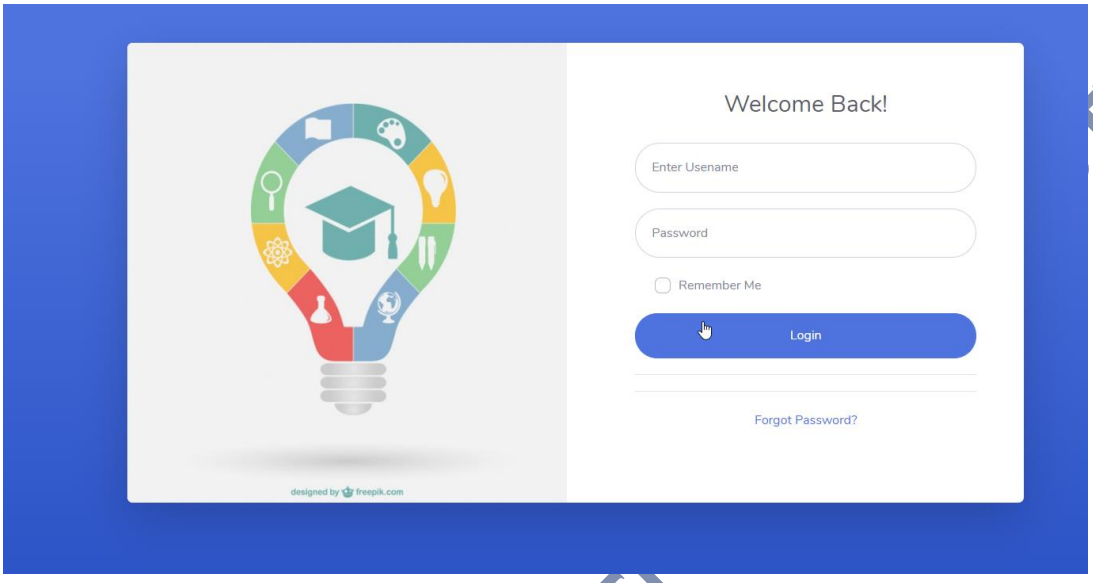

Figure 16:Login Page

AUTOMATIC ATTENDANCE SYSTEM

4.2.2.2 Forgot Password User Interface

If the user forgets his password, he will be redirected to this page by his request then he enters the email address associated with his account, and an email for password reset is sent only if such an email is found in the database.

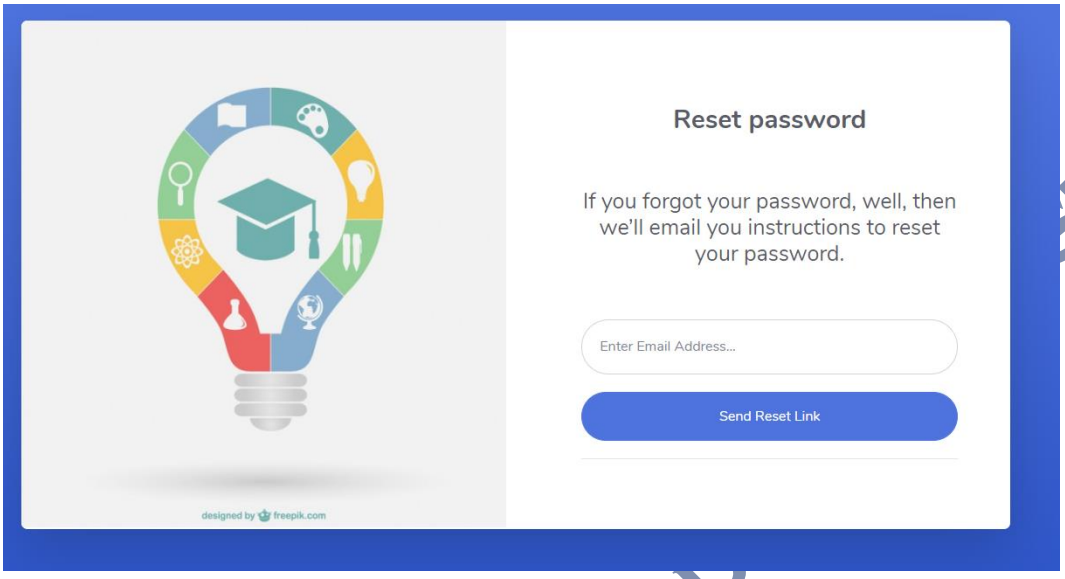

Figure 17:Password Reset Page

4.2.2.3 User List (Admin)

In the administrator user interface, the list of system users will be shown for the administrator to manage system users and add images to students

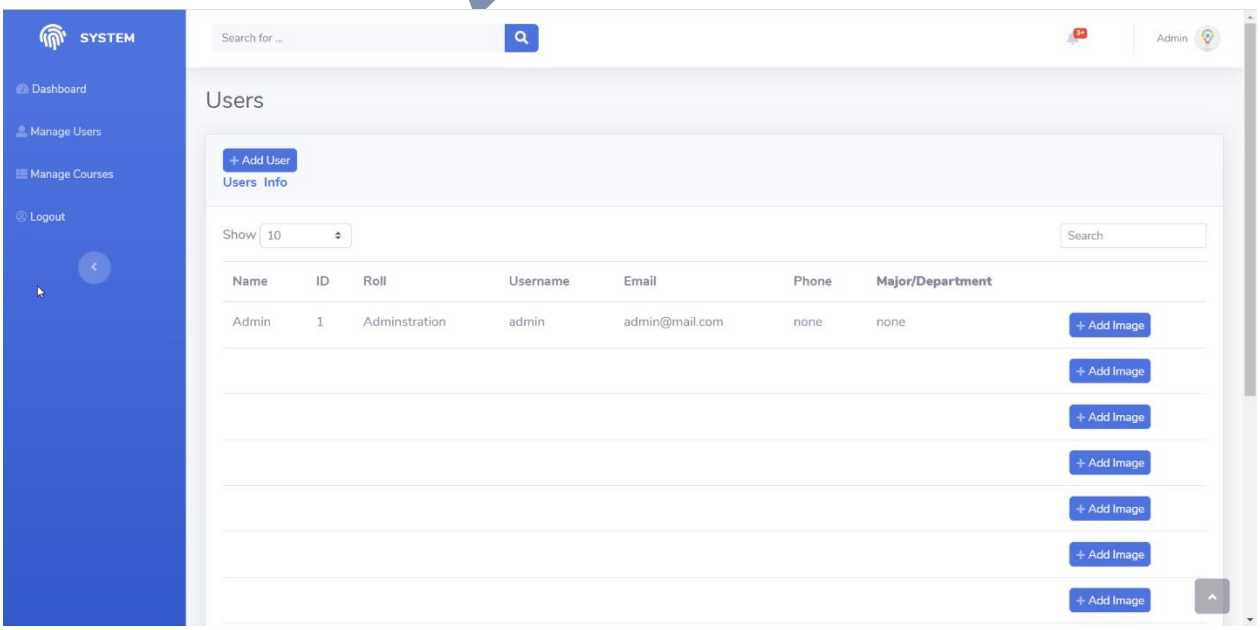

Figure 18:User List Page

4.2.2.4 Add New User(Admin)

the page where admin can add new users' details

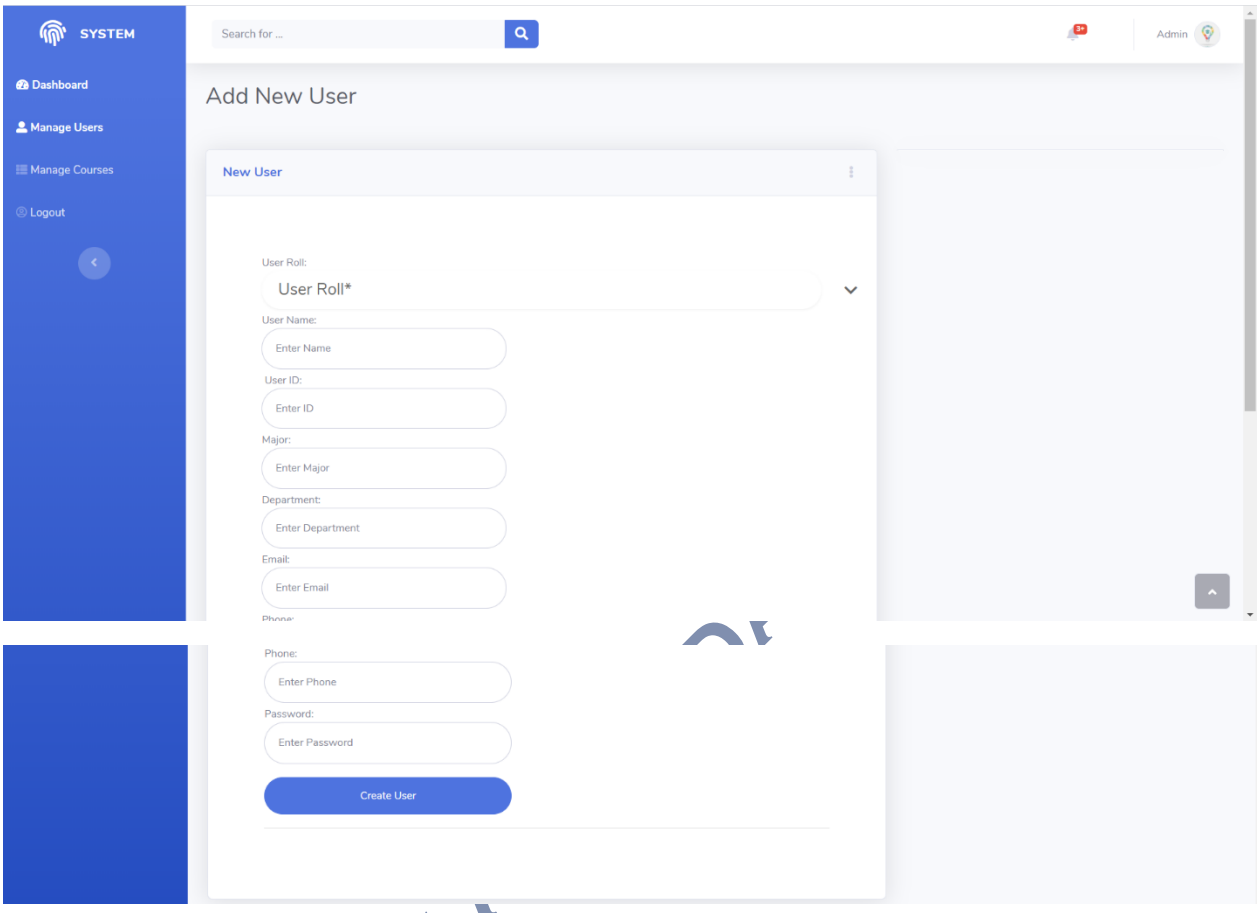

Figure 19:Add New User Page

AUTOMATIC

4.2.2.4 Add User Image (Admin)

The page where admin adds students' images to the face recognition system.

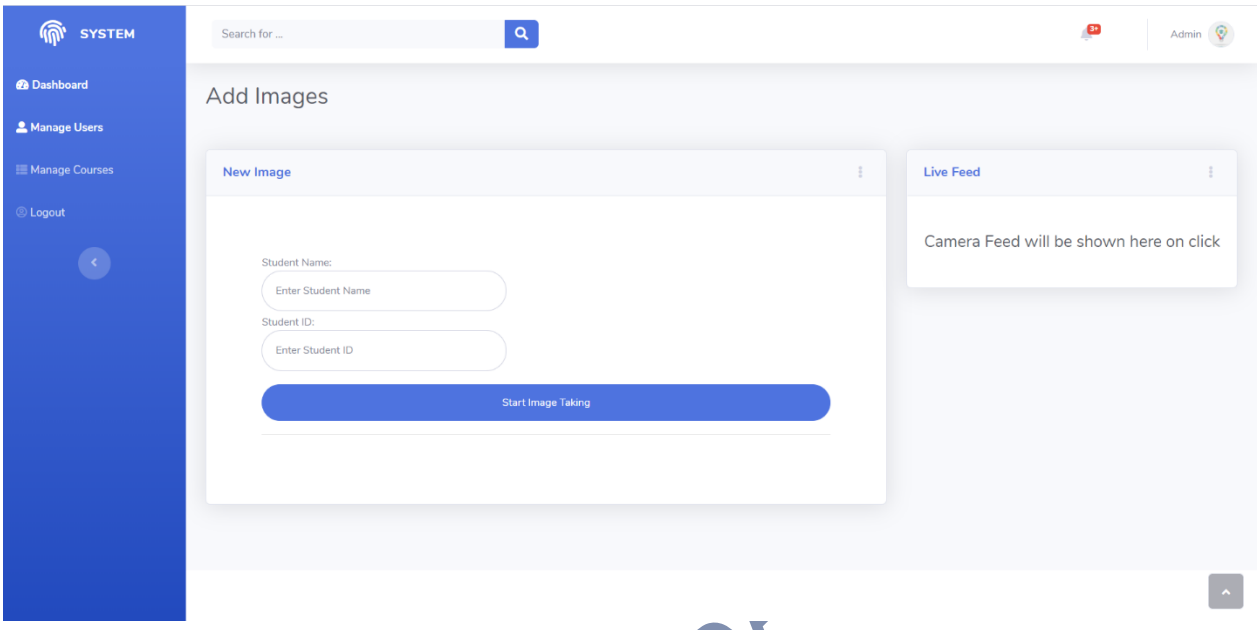

Figure 20:Add User Image Page

4.2.2.5 Courses Add(Admin)

The page where admin adds a new course to the system.

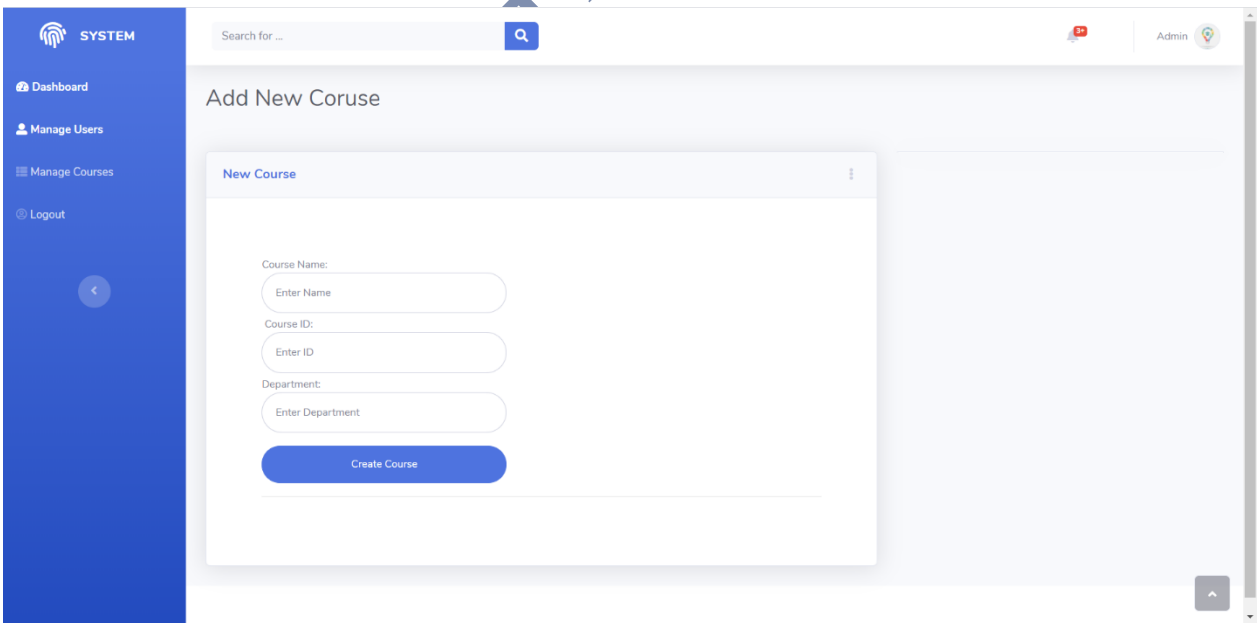

Figure 21:Adding Courses

4.2.2.6 Courses List (Admin)

Page to view courses list

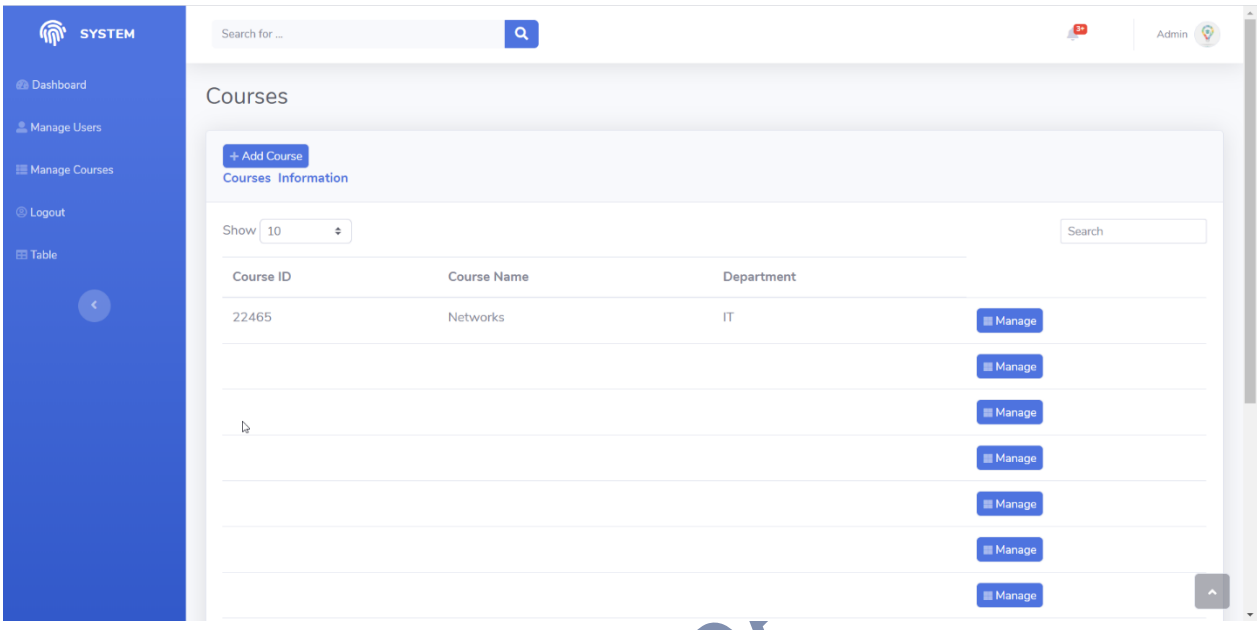

Figure 22:Course List Page

4.2.2.7 Course Management (Admin)

The page where admin can manage all courses in the system

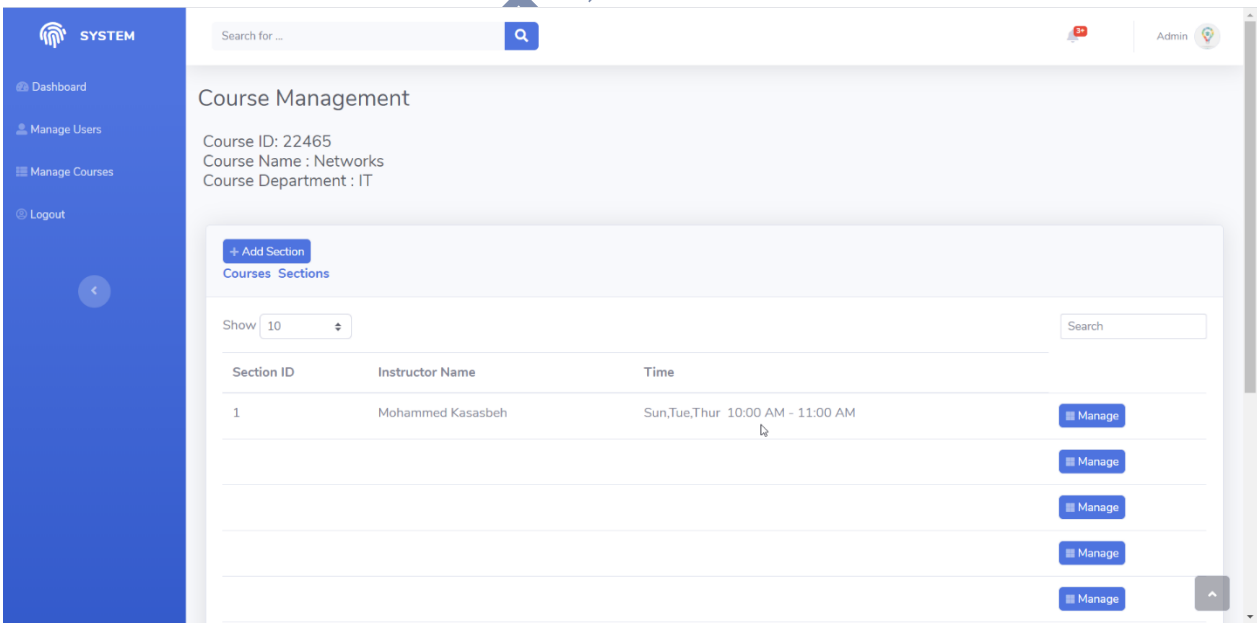

Figure 23:Course Management Page

4.2.2.8 Add Section (Admin)

The page where admin can add a new section to a course and assign an instructor and add students

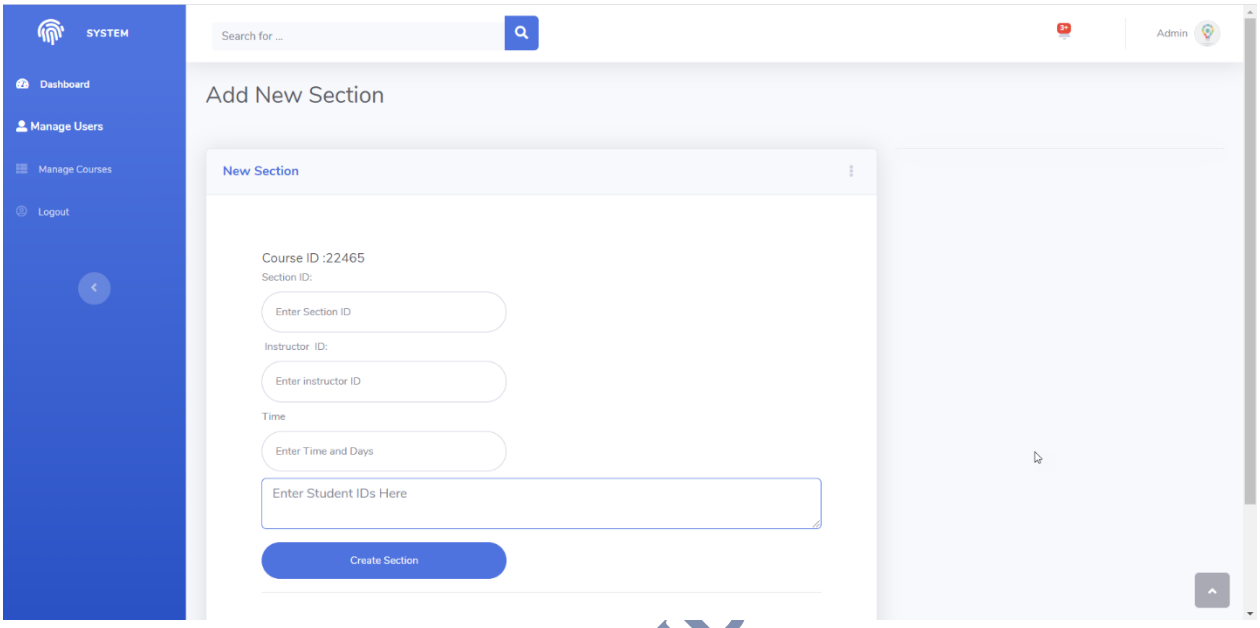

Figure 24:Add Section Page

4.2.2.9 Section Management (Admin)

The page that admin can manage each course sections from

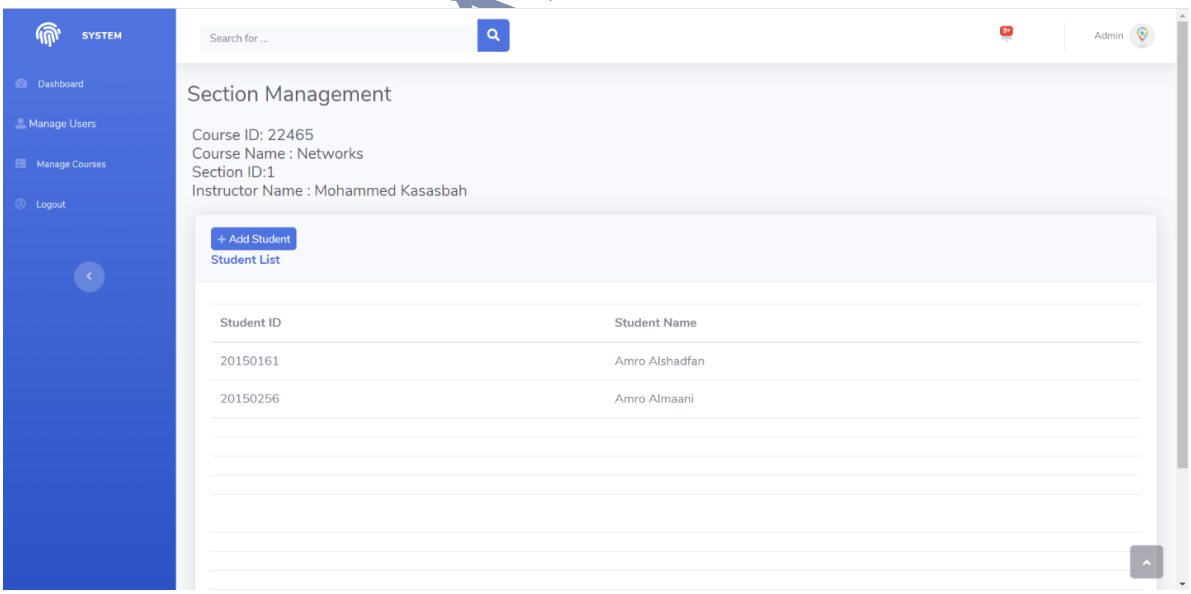

Figure 25:Section Management Page

4.2.2.10 Instructor Dashboard

The first-page instructor sees after log in where he /she can take attendance, update grades and take attendance and send notifications to students

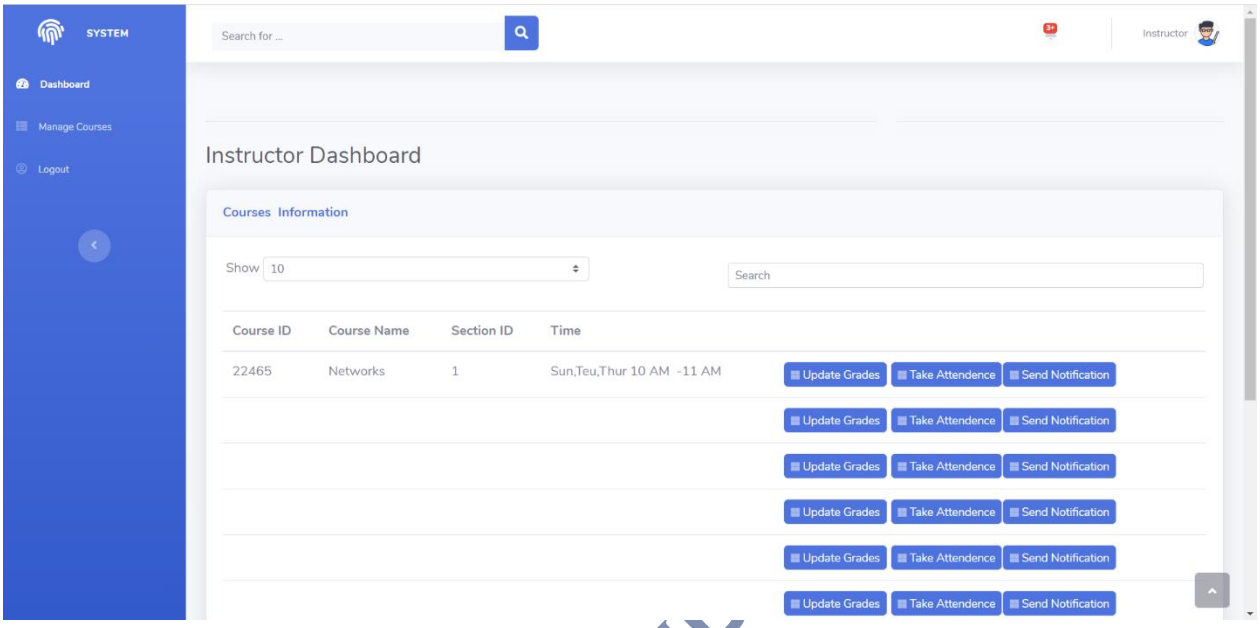

Figure 26:Instructor Dashboard Page

AUTOMATIC ATTENL

4.2.2.11 Take Attendance (Instructor)

The page that instructor takes attendance of students through face recognition

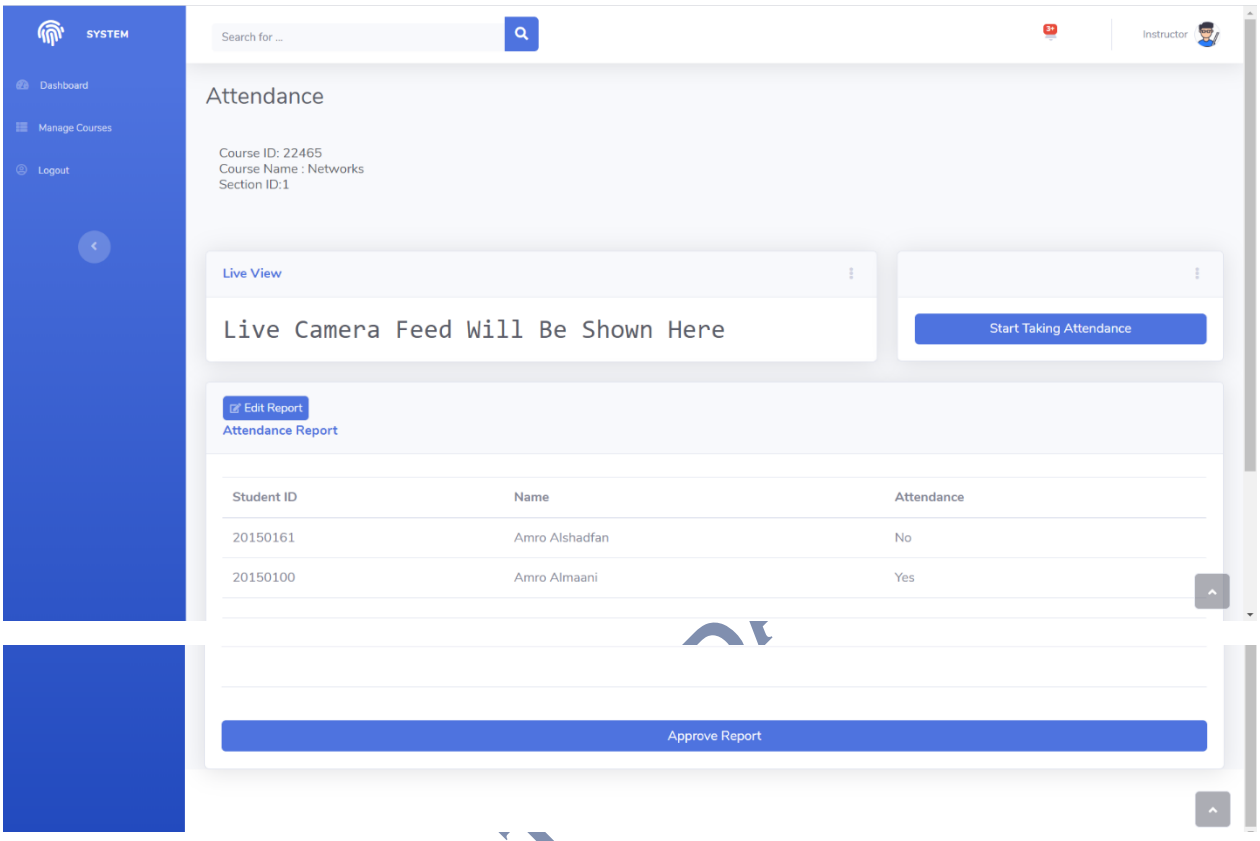

Figure 27:Taking Attendance Page

AUTOMATIC A

4.2.2.13 Update Grades (Instructor)

The page where the instructor enters students' grades for each course section assigned to him/here.

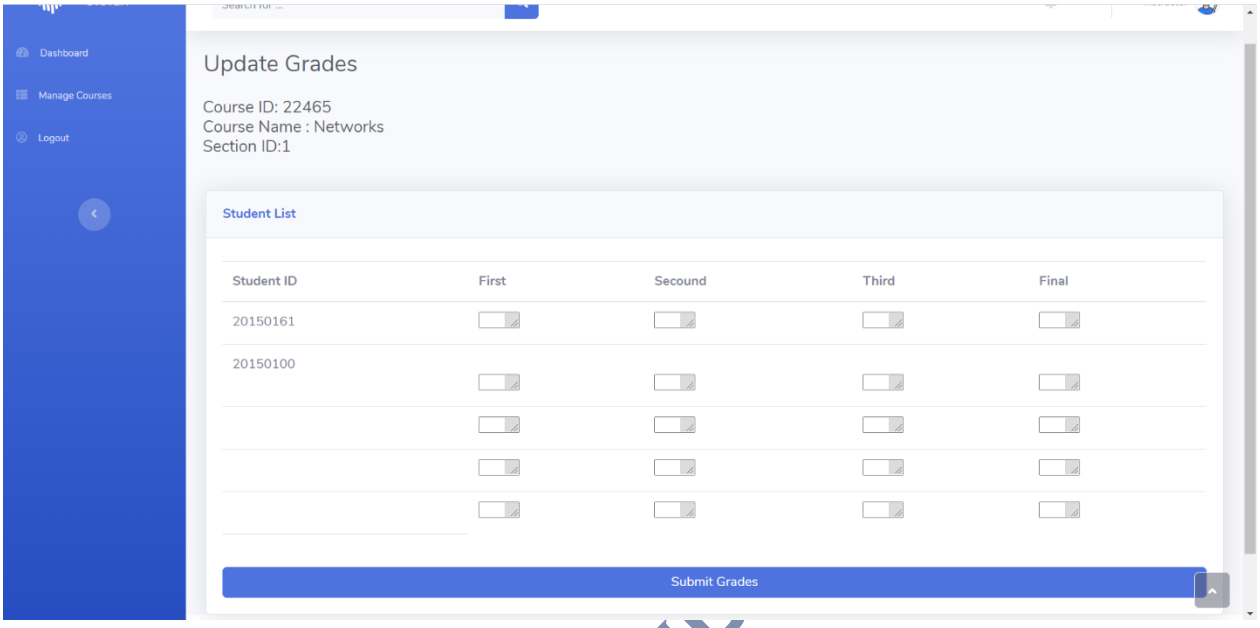

Figure 28:Update Grades Page

4.2.2.14 Send Notification (Instructor)

The page that instructor uses to send notifications to students in each

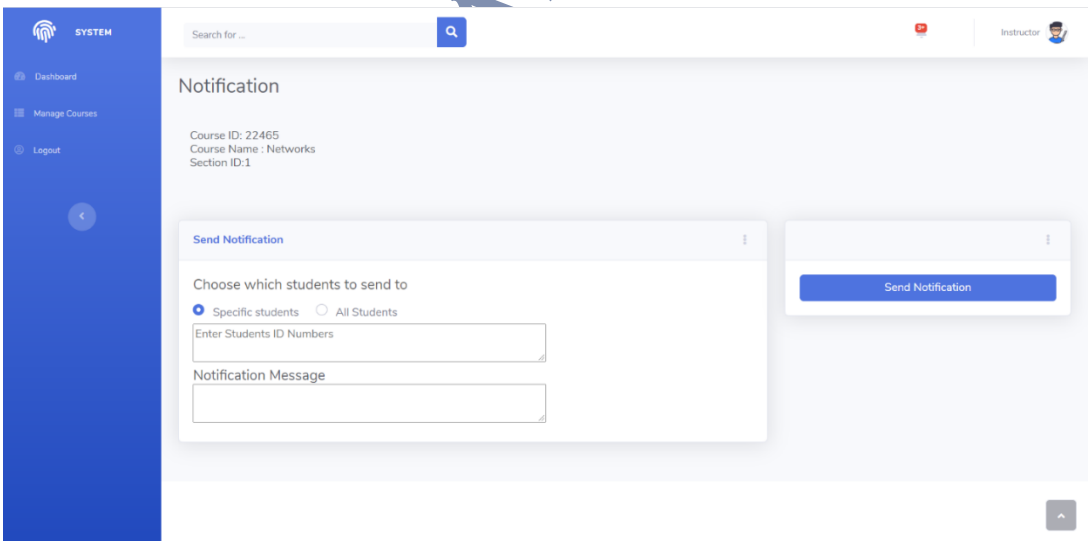

Figure 29:Send Notification Page

4.2.2.15 Student Dashboard

The page that the student is redirected to after login in which he /she can view notifications and attendance and grades for each course they are registered in.

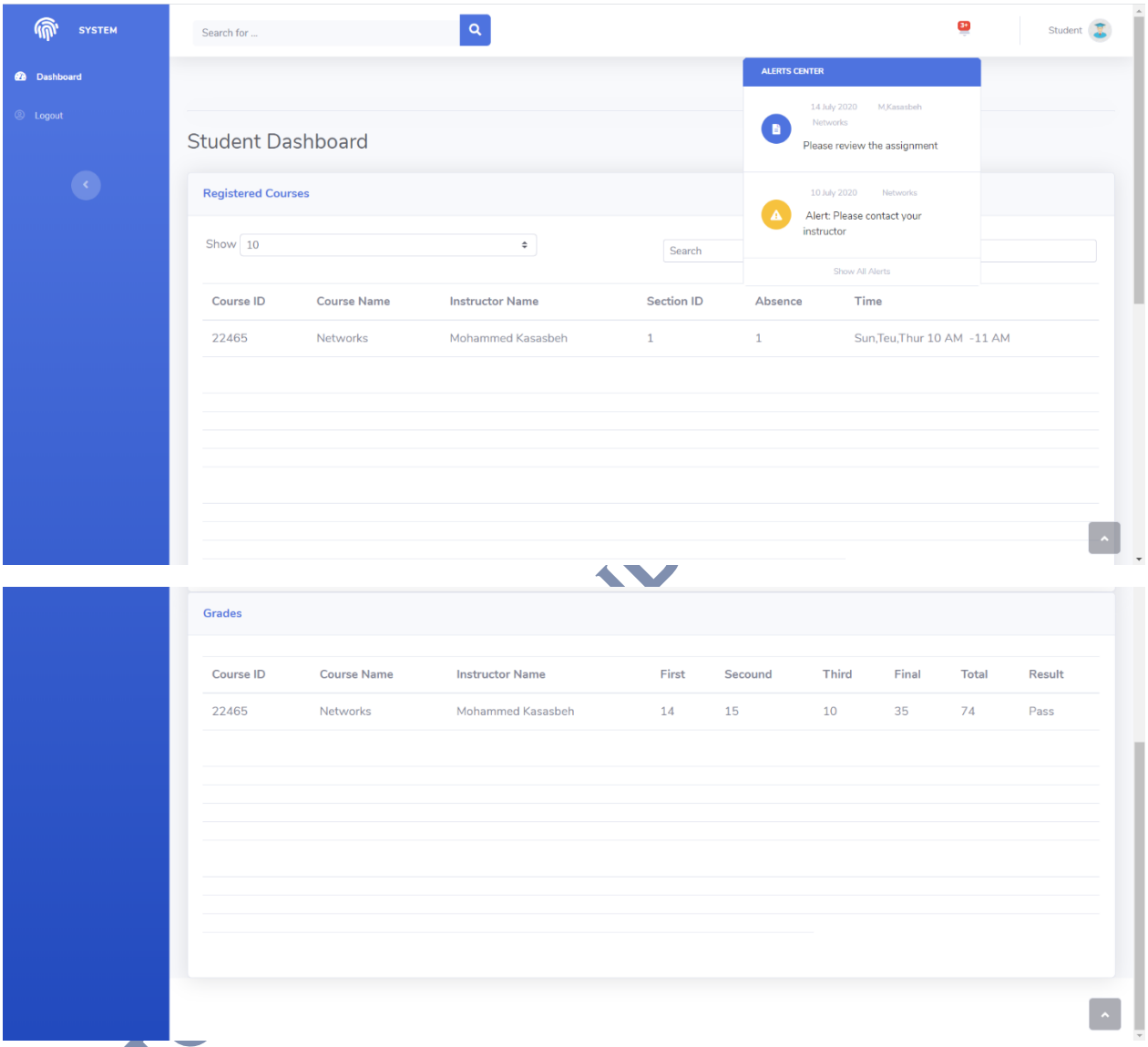

Figure 30:Student Dashboard

4.2.3: Database Design:

This section provides a prototype of the database tables and columns that we expect to be using during the implementation phase of our system and not a final database structure as more tables and columns will be used during the implementation phase to connect our tables and add more features.

4.2.3.1: Users Table:

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	**Username**	varchar(10)	utf8_general_ci		No	*None*		
☐	2	**Password**	varchar(16)	utf8_general_ci		No	*None*		
☐	3	**Roll**	text	utf8_general_ci		No	*None*		

Figure 31:Users Table

4.2.3.2: Administrator Table:

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	**username**	varchar(10)	utf8_general_ci		No	*None*		
☐	2	**Admin_ID**	varchar(8)	utf8_general_ci		No	*None*		
☐	3	**Admin_name**	text	utf8_general_ci		No	*None*		
☐	4	**Email**	varchar(50)	utf8_general_ci		No	*None*		
☐	5	**Phone**	int(20)			No	*None*		

Figure 32: Administrator Table

4.2.3.3: Instructor Table:

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	**username**	varchar(10)	utf8_general_ci		No	*None*		
☐	2	**Instructor_ID**	int(9)			No	*None*		
☐	3	**Instructor_Name**	text	utf8_general_ci		No	*None*		
☐	4	**Department**	text	utf8_general_ci		No	*None*		
☐	5	**Email**	varchar(50)	utf8_general_ci		No	*None*		
☐	6	**Phone**	int(20)			No	*None*		

Figure 33:Instructor Table

4.2.3.4: Student Table:

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	**username**	varchar(10)	utf8_general_ci		No	*None*		
☐	2	**Student_ID**	int(8)			No	*None*		
☐	3	**Student_Name**	text	utf8_general_ci		No	*None*		
☐	4	**Major**	text	utf8_general_ci		No	*None*		
☐	5	**Year**	date			No	*None*		
☐	6	**Email**	varchar(50)	utf8_general_ci		No	*None*		
☐	7	**Phone**	int(20)			No	*None*		

Figure 34:Student Table

4.2.3.5: Course Table:

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	**Course_ID**	int(10)			No	*None*		
☐	2	**Course_Name**	varchar(30)	utf8_general_ci		No	*None*		

Figure 35:Course Table

4.2.3.5: Section Table:

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	**course_ID**	int(6)			No	*None*		
☐	2	**Section_ID**	int(2)			No	*None*		
☐	3	**Instructor_ID**	int(9)			No	*None*		
☐	4	**Time**	varchar(50)	utf8_general_ci		No	*None*		

Figure 36:Section Table

Chapter 5
Implementation

5.1 Programming Languages

The system consists of many components connected and to achieve the best results we used the most appropriate language at our disposal for each function to be implemented in the desired form.

5.1.1 Front-End

Since our system is a web-based system the front-end is designed using Html5, CSS, JavaScript, and Bootstrap and Django handled the connection between the front-end and back-end through HTTP requests and JSON formatted data.

5.1.2 Back-End

Because our project implements attendance using facial recognition which requires machine learning and image processing activities we were forced to use python as a language in our development of the back-end and for python to work in a web environment we had to use Django framework or Flask framework which handles python functions for the back-end of the webserver.

Database connections are handled by Django using custom implemented modules and connections are opened and closed automatically based on HTTP requests

AUTOMATIC ATTENDANCE SYSTEM

5.2 Tools And API

Since python is the main development language of the back-end this granted us access to the amazing community-made libraries of python which made our task easier to accomplish with less code and effort while maintaining the quality of the work.

In the table below you will find fundamental libraries/frameworks used in our project noting that we used other libraries also to accomplish the project.

Table 20:Tools And API

Library/Framework	Description
1. DLIB	DLIB provides a pre-trained facial detector that is used throughout the facial recognition process to keep track of faces in real-time.
2. OpenCV	OpenCV provides a very big and widely used computer vision library that is used in our project to grab frames from the client-side and get them ready for facial recognition
3. Pillow	Pillow is an image processing library that is used to perform image processing on our dataset and through the face recognition process
4. Cmake	Cmake is a necessary library to run face recognition libraries in python
5. Face-recognition	Is responsible for managing the facial recognition models and overall process
6. Numpy	Provides the ability to do complex mathematical operations and deals with arrays which is a fundamental part of image processing and facial recognition
7. Rest framework and Django framework	Rest and Django manage the web API where Rest manages the authentication and Django handles user requests and python operations

AUTOMATIC ATTENDANCE SYSTEM

5.3 General Implementation Overview

In the front-end, the website consists of 32 pages such as login, dashboard, user management course and section management, attendance and grades management pages, and many other pages are managed by Django back-end and developed using a bootstrap studio which is a front-end development tool that uses Bootstrap, HTML5, Javascript and CSS, and PyCharm is a python development framework which is used to develop 90% of the back-end code

To measure the lines of code we used a python library called pygount which counts the lines of code implemented in the project which is 55,668 lines of code in total until now.

5.4 Main Algorithms and code convention

Django handles the database by creating a connection to the assigned PostgreSQL database on request and keeping it opened until no further requests or the expiry of the access token and on the event of a request to read, update or delete a middleware model is checked to get user access token and identify if the user has enough privileges before the execution of queries to the database.

The database is created automatically from a pre-defined model.py file which has the basic structure of the database and entries are done through objects.

The face recognition dataset is created using an algorithm implemented through the libraries explained briefly in the figure below

AUTOMATIC ATTENDANCE SYSTEM

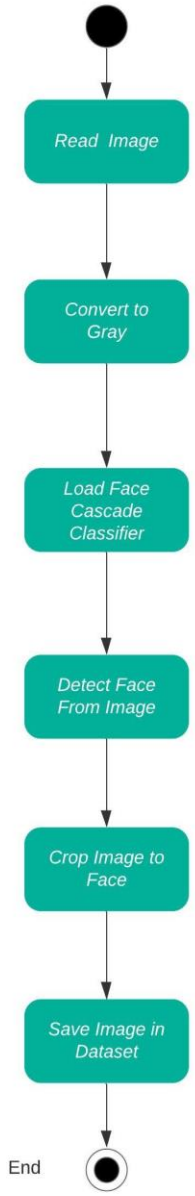

Figure 37: Dataset Creation Algorithm

AUTOMATIC ATTENDANCE SYSTEM

And the same algorithm is used for face recognition attendance but its expanded to fit the technical needs of facial recognition which is explained in the figure below.

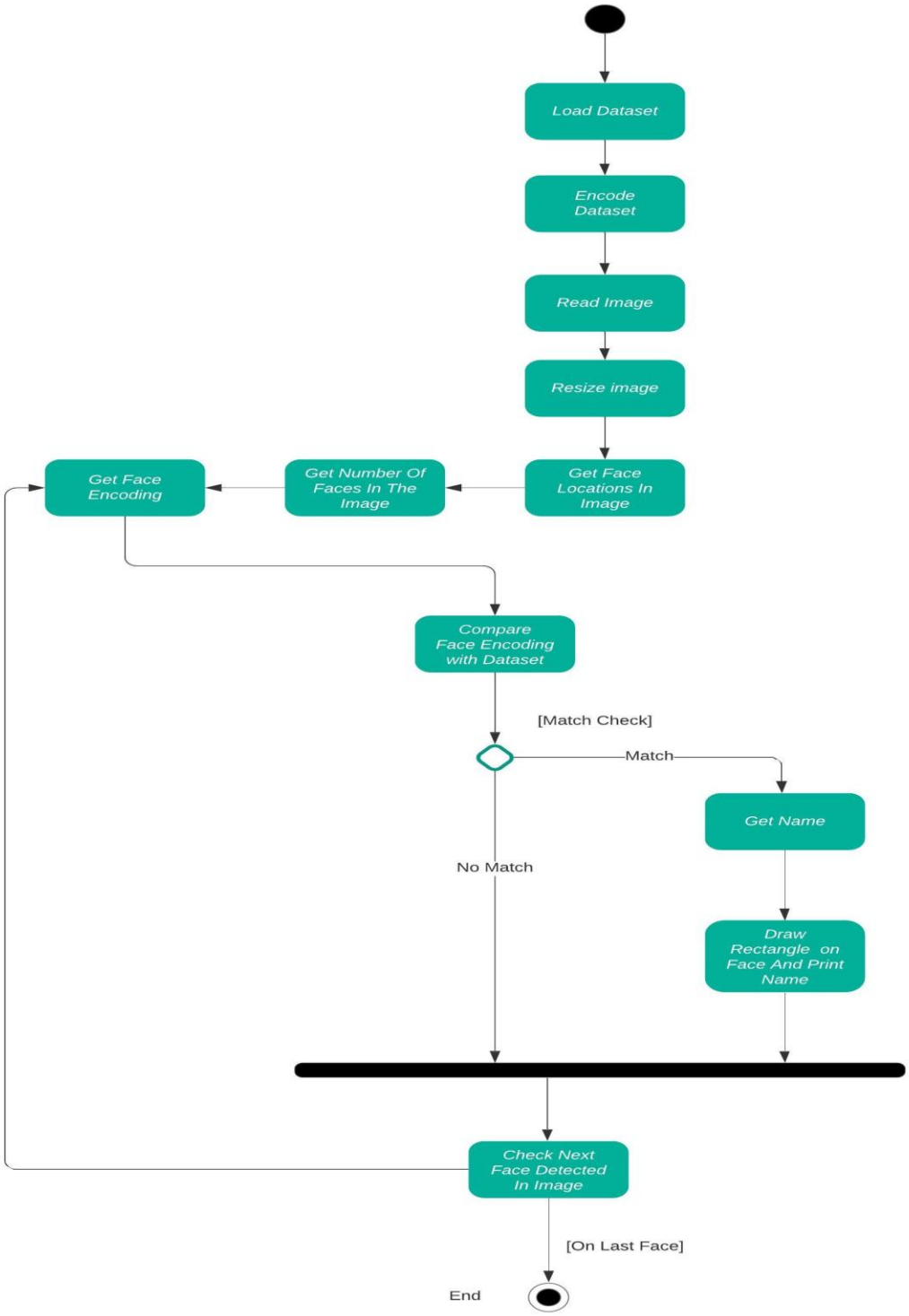

Figure 38:Face Recognition Algorithm

5.4.1 User Creation and editing

In the definition of the user in the 'models, it shows that the user takes 4 custom inputs which are user type, user id, phone, and major noting that Django creates fields like username, password, email, and full name by default using its default model and adds custom fields from our custom models file

```
class CustomUser(AbstractUser):
    USER_TYPES = ((1, "Admin"), (2, "Instructor"), (3, "Students"))
    user_type = models.CharField(max_length=10, default="1", choices=USER_TYPES)
    user_id = models.IntegerField(default=0)
    phone = models.CharField(max_length=12, default='000-000-0000')
    major = models.CharField(max_length=255, default='')
```

Since we created the adding user and editing user in the same page if we were creating a user a “0” value is passed to the function through the URL and a new user object is created then values are filled through requesting POST from the HTML page

```
user = None
if int(id) == 0:
    user = CustomUser()
```

And if the value of id passed through URL is not “0” the user object is created matching received id and information is fetched from the database and resent through HTTP response to the HTML page for display.

If the user decides to edit user information new values are taken through the POST method from HTML and the user is reinputted in the database using the user object

```
user.user_id = request.POST.get('user_id')
user.username = request.POST.get('username')
user.first_name = request.POST.get('first_name')
user.last_name = request.POST.get('last_name')
user.user_type = int(request.POST.get('user_type'))
user.email = request.POST.get('email')
user.phone = request.POST.get('phone')
user.major = request.POST.get('major')
user.save()
```

5.4.2 Course creation

Each course has a course id and name as defined in our models file

```
class Course(models.Model):
    course_id = models.CharField(max_length=255, default='0')
    course_name=models.CharField(max_length=255)
    objects=models.Manager()
```

Upon a request to create a course id and course name are sent from the HTML page to the back end through POST then a Course object is created and saved in the database

```
'''
course_name = request.POST.get("course_name")
course_id = request.POST.get("course_id")
try:
    course = Course(course_name=course_name, course_id=course_id)  # creating
course object
    course.save()
    messages.success(request, "Course <%s> successfully added! " % course_name)
```

5.4.3 Section creation

After course creation the admin is required to add sections to each course each section takes a section id and section name and time includes the days each section is in and the assigned instructor for the section and the students id's to enroll in the section and the section connects to the instructor id and course id automatically and the data is sent to the back-end through POST

```
section_name = request.POST.get('section_name')
instructor =
Instructor.objects.get(id=request.POST.get('instructor_id'))
course = Course.objects.get(id=request.POST.get('course_id'))
start_time = request.POST.get('start_time')
end_time = request.POST.get('end_time')
days = request.POST.getlist('day')
user_ids = request.POST.get('student_ids').split(',')
```

And a section object is created and saved to the database

```
section = Batch()
section.batch_name = section_name
section.course_id = course
section.instructor_id = instructor
section.start_time = start_time
section.end_time = end_time
section.days = ','.join(days)
section.save()
```

Then a student object is created for each student id inputted and the section id is appended in the section array of each student to register them in the section and the object is saved to the database

```
for user in CustomUser.objects.filter(user_id__in=user_ids, user_type=3):
    students.append(Students.objects.get(user_id=user.id))

for student in students:
    student.batch_id.add(section)
```

5.4.4 Adding student to the image dataset

On admin request user id is posted to the function for adding student face to the dataset then a user object is created and the user information is fetched from the database and a student object is created .

```
    userid=request.POST.get("userid")
    user=CustomUser.objects.get(id=userid)
    stu_obj = Students.objects.get(user=user)
```

then camera is started and frames are rendered then converted to grayscale then the haarcascade classifier responsible of detecting faces is loaded and the gray image is passed along with the scaling factor and other parameters to the cascade classifier to detect face in each frame and save it in faces .

```
cap = cv2.VideoCapture(0)
while True:
    ret, img = cap.read()
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faceCascade = cv2.CascadeClassifier(cv2.data.haarcascades +
"haarcascade_frontalface_default.xml")
    faces = faceCascade.detectMultiScale(
        gray,
        scaleFactor=1.3,
        minNeighbors=3,
        minSize=(30, 30)
    )
```

After detection for each face detected a blue rectangle is drawn on the face and the frame cropped to the face detected and saved in JPG format named with the student id number along with the frame number

```
key = cv2.waitKey(1) & 0xFF
for (x, y, w, h) in faces:
    cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
    cv2.imshow("Webcam", img)
```

```
    cv2.imwrite((os.path.join(BASE_DIR, "ImageData/"))  + str(user.user_id)
+'.' + str(count) +".jpg",gray[y:y+h,x:x+w])
```

the system will keep taking photos of the person and saving them until the user clicks “s” on the keyboard obviously the more images taken the better the recognition is .

after clicking “s” a success message is displayed and the student facetaken attribute is set to true

```
if key == ord("s"):
    messages.success(request, "Face added successfully")
    stu_obj.faceTaken = True
    stu_obj.save()
```

example of image taken:

Figure 39:Dataset image example

5.4.5 Taking attendance of a section

While logged in as an instructor if the user chooses to take attendance section id is posted to the back end then a section object is created to fetch the information of the posted section along with an attendance object which takes the current date and section id

```
batchid = request.POST.get("batchid")
batch_obj = Batch.objects.get(id=batchid)
att = Attendance.objects.get_or_create(subject_id=batch_obj,
attendance_date=datetime.date.today())
attid = att[0].id
```

then three arrays are created to save images and student names and the attendance list and the list of all files in the dataset files are saved in an array

```
images = []  # for name of images with extention
imageNames = []  # for name of images without extention
studentMarked = []  # list of students already marked present
exactpath = os.path.join(BASE_DIR, 'ImageData')
myList = os.listdir(exactpath)
```

AUTOMATIC ATTENDANCE SYSTEM

as we mentioned before each file in the dataset is named according to the student id and picture number so after the list of files is saved in an array the array is scanned and students are fetched one by one and checked if the student is registered in the section or not to avoid adding a non-registered student in the attendance report then if the student is registered is id is added in an array and the images of that student is read and added to another array

```
for cls in myList:

    userId1 = cls.split('.')[0]
    user = CustomUser.objects.get(user_id=userId1)
    std = Students.objects.get(user_id=user.id)
    batch = std.batch_id.get(id=batchid)

    # if this student is not enrolled in this secion, except from attendance
    if batch == None:
        continue
    imageNames.append(userId1)

    curImg = cv2.imread(f'{exactpath}/{cls}')
    images.append(curImg)
```

then the images are sent to a function that reads the encoding of each image in the array of images we just created where each image is converted to BGR then encodings of faces are extracted to a list and sent back to the attendance function

```
encodeListKnown = findEncodings(images)
```

```
def findEncodings(images):
    '''
    creating a list of encodings for all images
    '''
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encodings = face_recognition.face_encodings(img)
        if len(encodings) > 0:
            encodeImg = encodings[0]
            encodeList.append(encodeImg)
        else:
            continue
    return encodeList
```

then capturing video from camera it started and each frame is resized for faster processing and face locations and its encodings are saved then for each face in the frame then current encodings are compared with the encodings from the dataset using convolutional neural networks and a 0.6 tolerance

```
cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)  # capturing video

while (True):
    ret, img = cap.read()
    imgSmall = cv2.resize(img, (0, 0), None, 0.25, 0.25)  # resizing for
faster processing
    rgb_small_frame = cv2.cvtColor(imgSmall, cv2.COLOR_BGR2RGB)  # converting
image to RGB mode

    facesCurFrame = face_recognition.face_locations(rgb_small_frame)  #
getting face location in current image
    encodesCurFrame = face_recognition.face_encodings(rgb_small_frame,
                                                      facesCurFrame)
```

If a match is found user id is fetched form the array then a user and student object are created and the name of the identified student is fetched and a rectangle is drawn around the face and name is displayed on the video feed and the identified student is marked as attendant

```
userId = imageNames[matchIndex]  # this is student id

user = CustomUser.objects.get(user_id=userId)
std = Students.objects.get(user_id=user.id)
stdId = std.id
name = user.get_full_name()  # this is student name
y1, x2, y2, x1 = faceLoc  # locations of face detected
y1, x2, y2, x1 = y1 * 4, x2 * 4, y2 * 4, x1 * 4  # resizing image back to
normal size
cv2.rectangle(img, (x1, y1), (x2, y2), (255, 0, 0),
              2)  # creating a rectangle around the detected face
cv2.putText(img, name, (x1 + 6, y2 - 6), cv2.FONT_HERSHEY_COMPLEX, 1, (51,
255, 255),
            2)  # for showing student name on the rectangle above
if stdId not in studentMarked:  # marking attendance if not already marked
    studentMarked.append(stdId)  # adding to marked list
    markAttendance(student_id=stdId, att_id=attid)  # marking attendance
```

and this process goes on until the user clicks “s” meaning that all students have been seen and closing the video feed

```
cv2.imshow("Webcam", img)
key = cv2.waitKey(1) & 0xFF  # for waiting till key 's' is pressed
if key == ord("s"):
    cap.release()
    cv2.destroyAllWindows()
    break
```

Taking attendance face recognition example:

Figure 40:Face recognition example

Note: the system may not always be able to recognize faces while wearing a mask it's a matter of dataset size of each person.

5.5 Requirements Coverage

Out of 11 functional requirements specified in chapter 3 section 3.3 we have successfully implemented 10 functional requirements during this project. This results in a 90% coverage of functional requirements we planned to implement in our project.

Below is a table summarizing which requirements got implemented and which were not:

Table 21:Requirement Coverage

Requirement ID	Requirement Name	Requirement Detail	Priority	Implemented?
FR1	Register new user	To register a new user to the system and add it to the database	Essential	Yes
FR2	Sign in	To allow the user to log in to his/her account login the system	Essential	Yes
FR3	Forgot password	Recovering user password in case he/she forgets and setting up a new password for his/her account	Essential	Yes
FR4	Add student images	Adding students' images to the database to prepare for face recognition	Essential	Yes
FR5	Add courses and sections	Adding new courses and sections to the system and add students to the section and assign an instructor to each section	Essential	Yes

AUTOMATIC ATTENDANCE SYSTEM

FR6	View assigned courses	For instructors to view the courses they are teaching, and the students registered in each section	Essential	Yes
FR7	Take attendance	To take attendance of students for each lecture using face recognition	Essential	Yes
FR8	Update and add grades	Managing grades for each course	Essential	Yes
FR9	Generate statistics	Generating a report for each course section that shows instructor how attendance affects grades	Recommended	No
FR10	Send Notifications	The ability to send notifications to the students	Recommended	Yes
FR11	View registered courses attendance and grades and notifications	View registered courses attendance and grades and notifications	Essential	Yes

AUTOMATIC ATTENDANCE SYSTEM

Chapter 6
Testing

6.1 Testing Approach

Since testing is an essential part of the development and every system must go through tests we created a clone database populated with dummy test data to avoid interference with our main database.

The system was tested using the Unit testing approach in which each function of the system is isolated and tested on its own to ensure that each part functions properly and to ensure full coverage of our system segments.

Unit testing was done through a normal client connection to ensure that what we are testing what will be used by the normal user, not a high privileged user.

Before starting unit testing, we reviewed each function to make sure that there are no syntax errors in the functions

Unit test cases were written to ensure 100% statement coverage, 100% path coverage and 100% decision coverage.

6.2 Testing Tools

Django framework has a pre implemented testing framework that we decided to use to test our project because it tests the system in-depth including URLs and database structure and main functions of the system and some front-end functions and code.

And to prevent the automated testing data from interfering with our main system data we created an isolated testing environment along with a database separate from the main database and we populated it with dummy data using the Django-seed framework library which generates test data according to specified data types .

Automated testing is done through 3 files test_models.py, test_views.py and test_urls.py each file containing special lines of code indicating steps and pre-conditions and post-conditions

In addition to using the automated testing framework provided by Django we also conducted manual tests on our system through the user interface throughout the development phase and the testing phase.

AUTOMATIC ATTENDANCE SYSTEM

6.3 Testing Sample

Below is a test sample of the conducted 12 manual tests and their outcome test report

Table 22:Test Sample report

Test Case ID	TC1
Test Title	User registration
Related Function	FR 1
Test Description	This test case tests the ability to register a new user of all types to the system which includes various subcases including: • Entering a non-existing user id and all required information correctly and matching data types and a password matching the required criteria along with its confirmation matching the inputted password. • Entering a non-existing user id with missing some required information and a password matching the required criteria along with its confirmation matching the inputted password. • Entering a non-existing user id and all required information correctly but not matching the data types in database structure and a password matching the required criteria along with its confirmation matching the inputted password. • Entering a non-existing user id and all required information correctly and matching data types and a password not matching the required criteria and a miss match in password confirmation • Entering an existing user id and all required information correctly and matching data types and a password matching the required criteria along with its confirmation matching the inputted password.
Pre-conditions	• Having a non-existing user id for the new user • Having the inputted data matching the required data types in the database • Having a Password matching the required criteria which is containing upper-case, lower-case, numbers and symbols in the password phase • Password confirmation must match the inputted password

AUTOMATIC ATTENDANCE SYSTEM

Test Steps	1. Admin logs in to the system using their account 2. Admin enters the user management page 3. Admin enters user creation page 4. Admin inputs a non-existing user id and other required information like first and last name , email , username , user type and phone number each in the required form and matching the required data type and inputs a correct password with a correct password confirmation 5. Admin inputs a non-existing user id with some other required information like first and last name , email , username , user type and phone number missing and inputs a correct password with a correct password confirmation 6. Admin inputs a non-existing user id and other required information like first and last name , email , username , user type and phone number with some not matching the correct form or data type and inputs a correct password with a correct password confirmation 7. Admin inputs a non-existing user id and other required information like first and last name , email , username , user type and phone number each in the required form and matching the required data type and inputs password in wrong formation or password confirmation not matching the inputted password 8. Admin inputs an existing user id and other required information like first and last name , email , username , user type and phone number each in the required form and matching the required data type and inputs a correct password with a correct password confirmation
Test Data Sample	User_id:20150265 First_name : Amr Last_name : Maani Username:20150265 Email: test@gmail.com Phone:0791234567 Major:CS User_type: student Password:Test1234$ Password Confirm:Test1234$

AUTOMATIC ATTENDANCE SYSTEM

Expected Results	• Step 4 executes successfully and a new user is created • Step 5 returns a missing information error and asks the admin to re enter the messing information • Step 6 returns an error stating that "expected string but received integers" and asks the admin to recheck the information inputted • Step 7 returns an error stating that passwords mismatch or password format is not correct • Step 8 returns an error that user already exists and asks the admin tor choose another user id
Post-condition	• User is created and shown in user list
Status	Pass

6.4 Tested Features

We have successfully tested all the functional requirements in this project except for the Generate statistics function because it was not implemented as we stated in chapter 5 section 5.5 of this documentation.

Table 23:Requirement Coverage

Requirement ID	Requirement Name	Requirement Detail	Priority	Tested ?
FR1	Register new user	To register a new user to the system and add it to the database	Essential	Yes
FR2	Sign in	To allow the user to log in to his/her account login the system	Essential	Yes
FR3	Forgot password	Recovering user password in case he/she forgets and setting up a new password for his/her account	Essential	Yes
FR4	Add student images	Adding students' images to the database to prepare for face recognition	Essential	Yes
FR5	Add courses and sections	Adding new courses and sections to the system and add students to the	Essential	Yes

AUTOMATIC ATTENDANCE SYSTEM

		section and assign an instructor to each section		
FR6	View assigned courses	For instructors to view the courses they are teaching, and the students registered in each section	Essential	Yes
FR7	Take attendance	To take attendance of students for each lecture using face recognition	Essential	Yes
FR8	Update and add grades	Managing grades for each course	Essential	Yes
FR9	Generate statistics	Generating a report for each course section that shows instructor how attendance affects grades	Recommended	No
FR10	Send Notifications	The ability to send notifications to the students	Recommended	Yes
FR11	View registered courses attendance and grades and notifications	View registered courses attendance and grades and notifications	Essential	Yes

AUTOMATIC ATTENDANCE SYSTEM

Chapter 7
Conclusions and Future Work

Face recognition technologies are recently being applied in various sectors worldwide such as schools, companies, universities , manufactures and government institutes since it's a very fast and secure way of identifying people and maintaining security and the usage of facial recognition has increased significantly during the Covid-19 global pandemic where other methods of id verification and logging requires physical contact which increases the health risks and threatens public safety facial recognition is a fast, secure and convenient solution .

In this project, we focused on implementing this technology to be used in schools and universities and we managed to cover most of the needs of a school or a university.

but this project can also be used in other places such as companies and governmental institutes.

Also, this project has a huge potential for improvement and a capacity for new features to be implemented in the future to be added to the system.

Firstly, the mentioned FR9 in chapter 3 section 3.3 which is responsible for generating statistical reports would greatly enhance the system capabilities by adding a very useful feature that benefits teachers and managers in evaluating the performance of their students or employees.

Secondly, transforming into a react Js user interface would not only make the system more elegant and easier to use but also it would work in a better way with the existing python Django base, noting that it was not implemented in the current phase due to the lack of experience in react JS and the tight schedule.

We also look forward to implementing a highly advanced facial recognition model that depends on machine learning and neural networks focusing more on using more artificial inelegance instead of the current OpenCV and mathematically related model while trying to maintain the speed of the system.

Lastly, implementing a mask detector that detects if the person accessing the institute is wearing a face mask or not will massively benefit any institute by helping in maintaining public safety and following government regulations especially during the covid-19 pandemic period.

AUTOMATIC ATTENDANCE SYSTEM

Appendix A

Users' Manual

This section provides a detailed user manual with screenshots to explain how the system can be used in a proper way.

The annotations are described as follows:

1) To login the user must access the homepage of the website

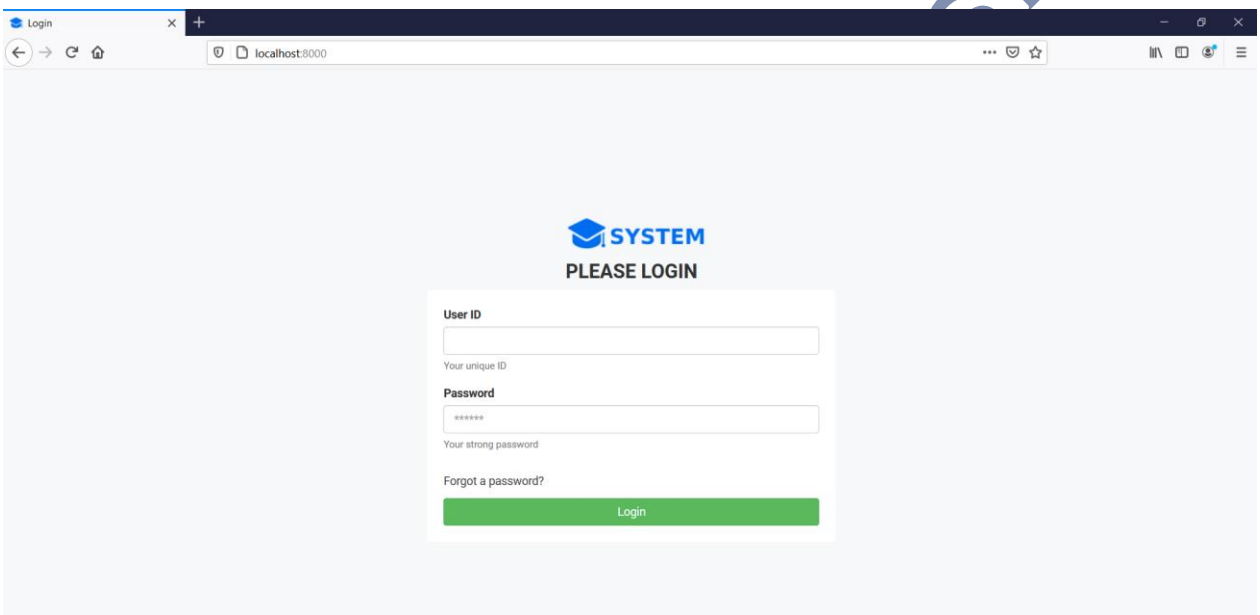

Figure 41:User manual – login page

AUTOMATI SYSTEM

2) The user should fill the credentials provided to him by the administrator

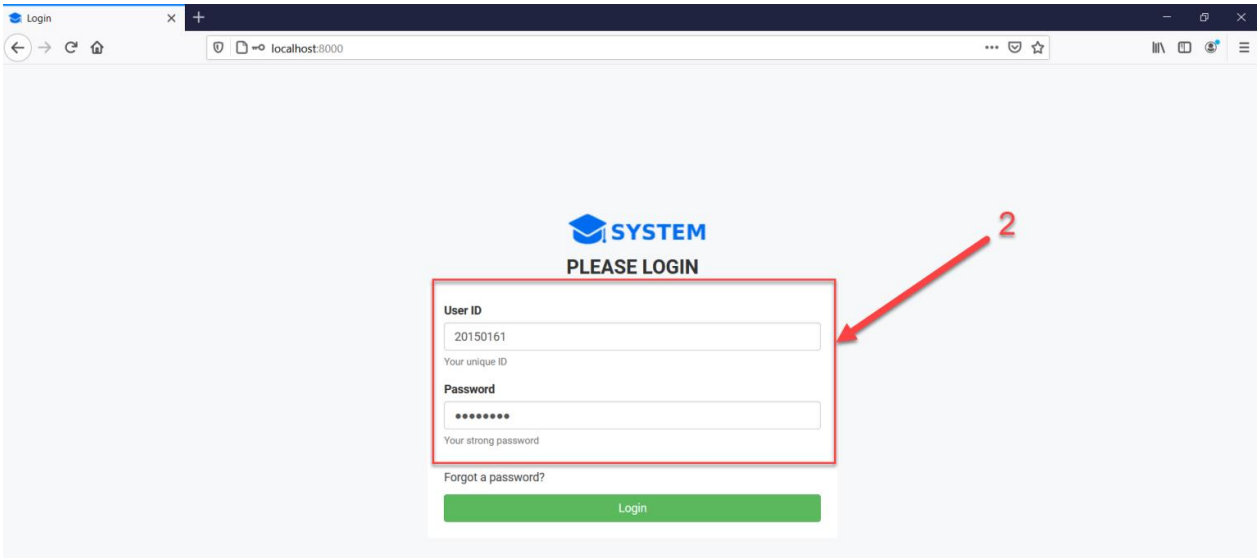

Figure 42: User manual – login page2

3) If the user forgot his/her password the user should click on the forget password then input the email associated with his account and a password reset link will be sent to his email

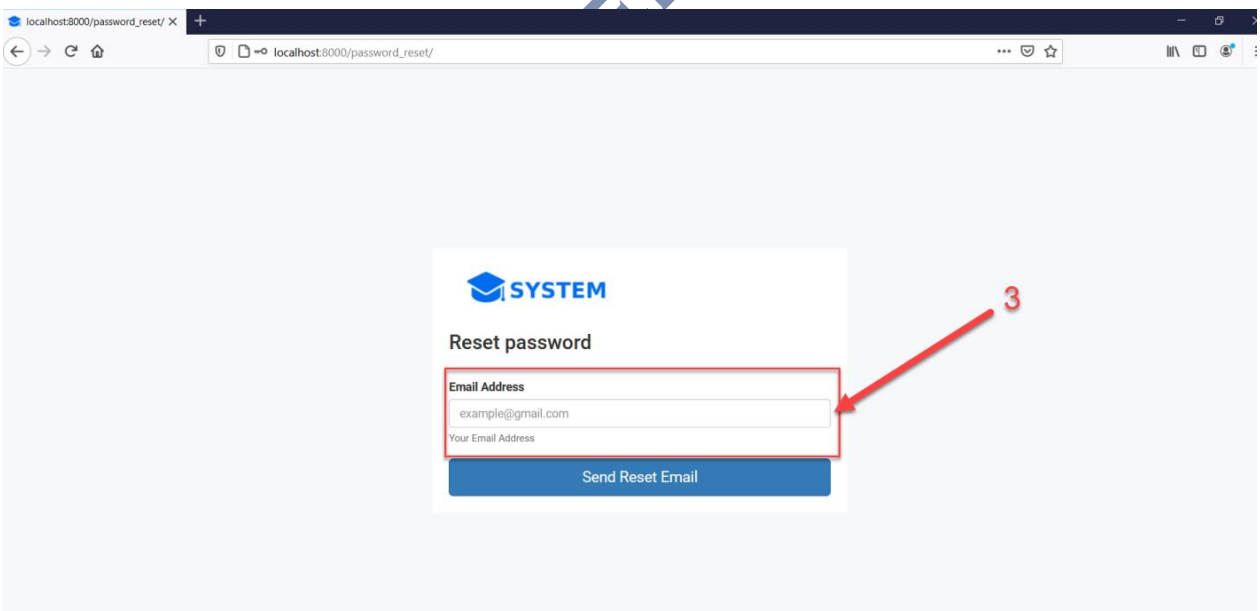

Figure 43: User manual – Password reset

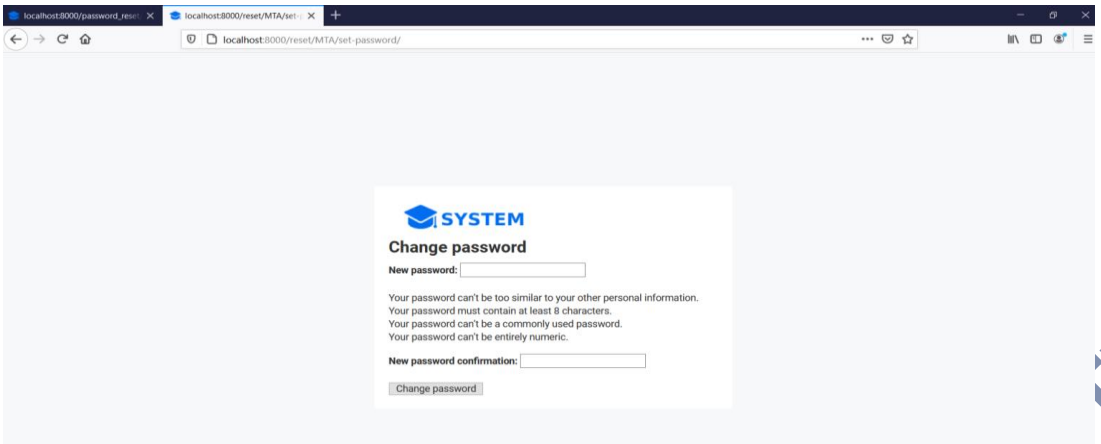

Figure 44: User manual – Password reset 2

4) Once logged in the user is shown a dashboard according to his/her user type (administrator, instructor, student)

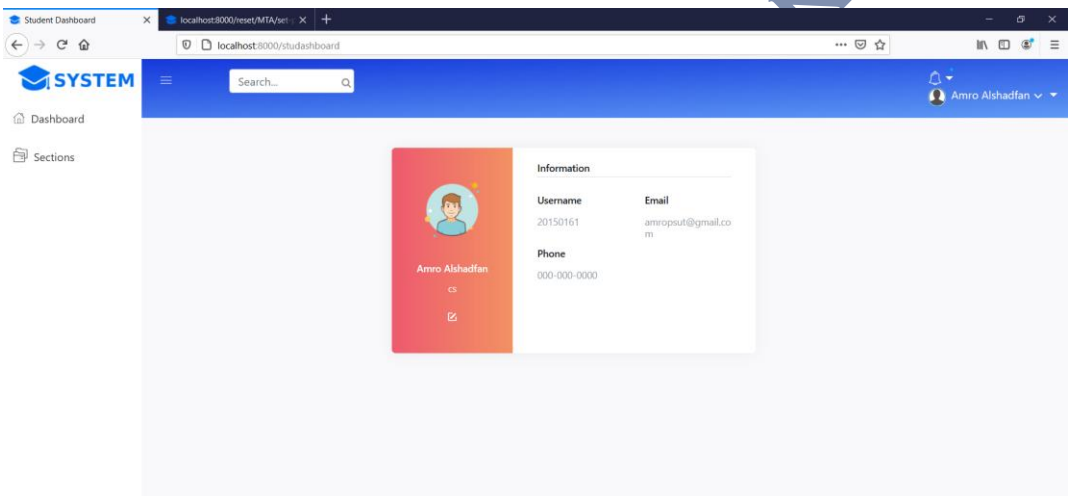

Figure 45 :User manual – Dashboard

AUTOMANCE SYSTEM

5) To logout from the system the user should click the logout button in the dropdown menu at the upper right corner of the screen

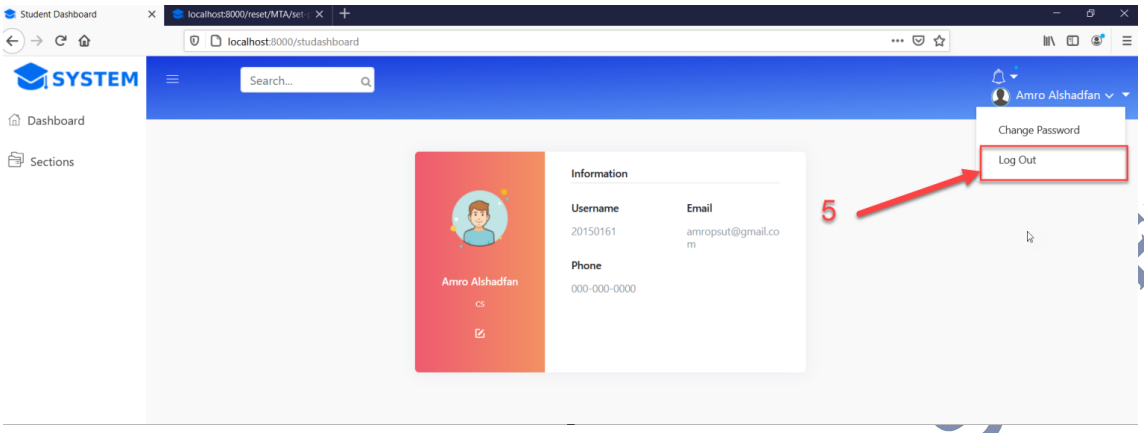

Figure 46: User manual – logout

6) To change user password user must click the change password button in the dropdown menu at the upper right corner of the screen

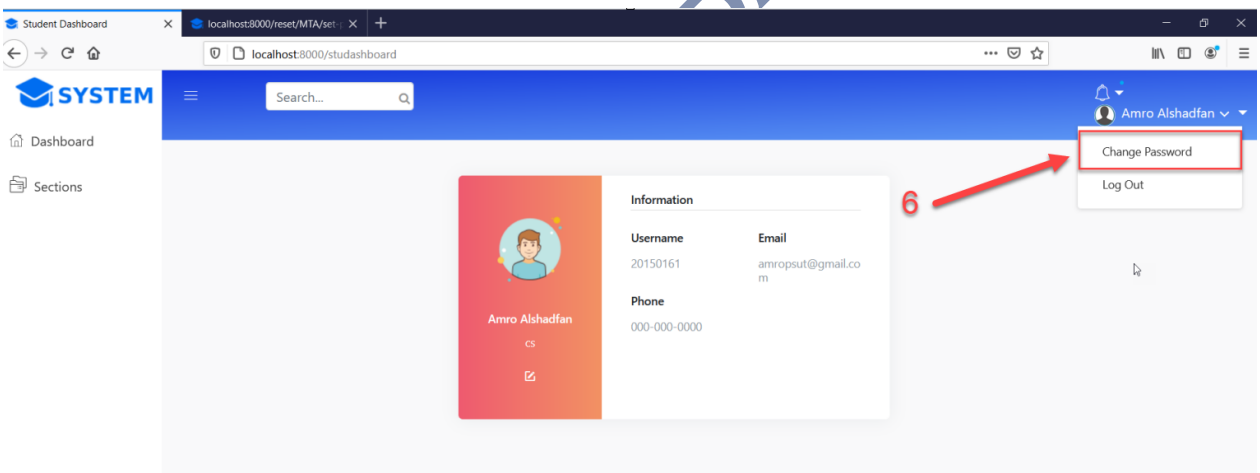

Figure 47: User manual – Password change 1

Then to change the password the user must enter the current password correctly and enter the new password two times and click update.

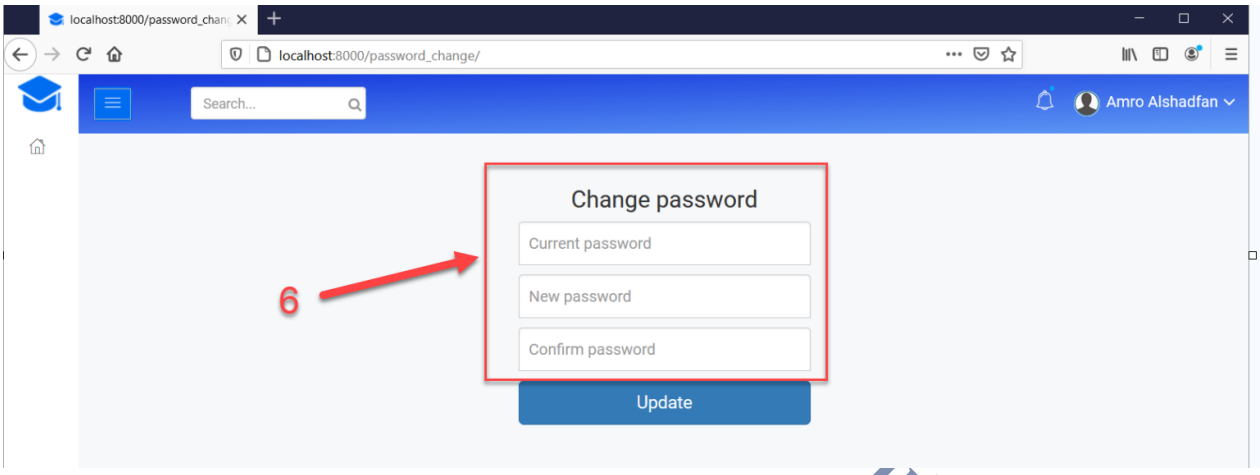

Figure 48: User manual – Password change 2

7) To add a new user the logged in user must be an admin then from the sidebar users tab is clicked.

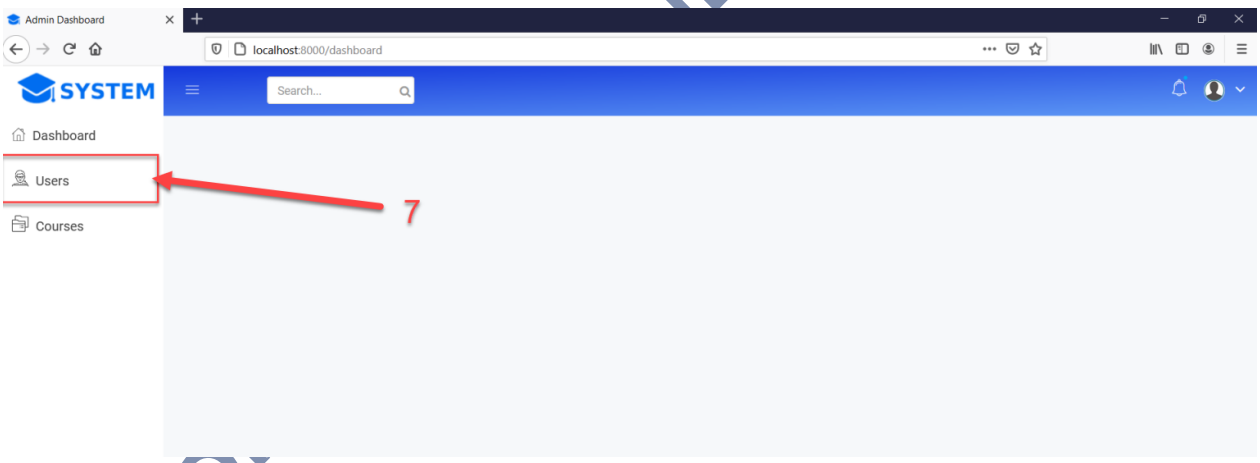

Figure 49 : User manual – Add new user 1

To add a user admin clicks the add user button then fills the required information then clicks add a new user then the user is created.

Figure 50 :User manual – Add new user 2

8) For student there is an option to add student face to the dataset from the users list click +face button on the desired student

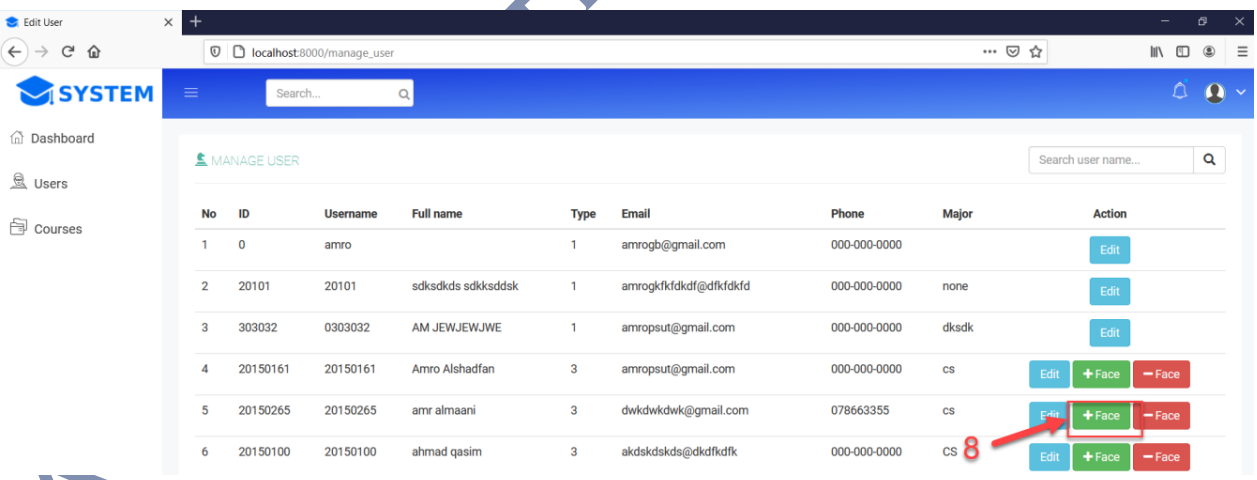

Figure 51 :User manual – Add student face 1

Then get the student close to the camera and click add face button and after that the student image is processed and saved 1 time every frame (the longer the better) until the user clicks "S" on the keyboard to end the process.

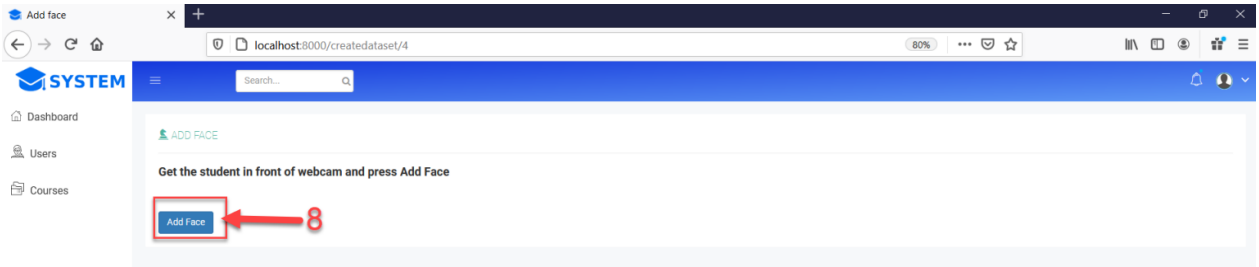

Figure 52 :User manual – Add student face 2

9) To add a new course from admin dashboard the admin user clicks courses tab from the

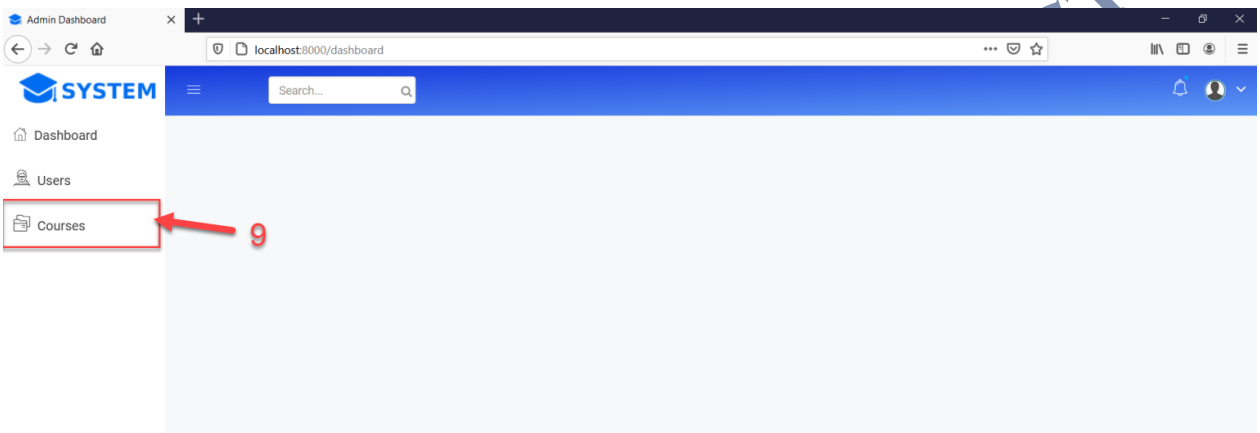

Figure 53 : User manual – Add course 1

Then on the courses list click add new course and fill the required information

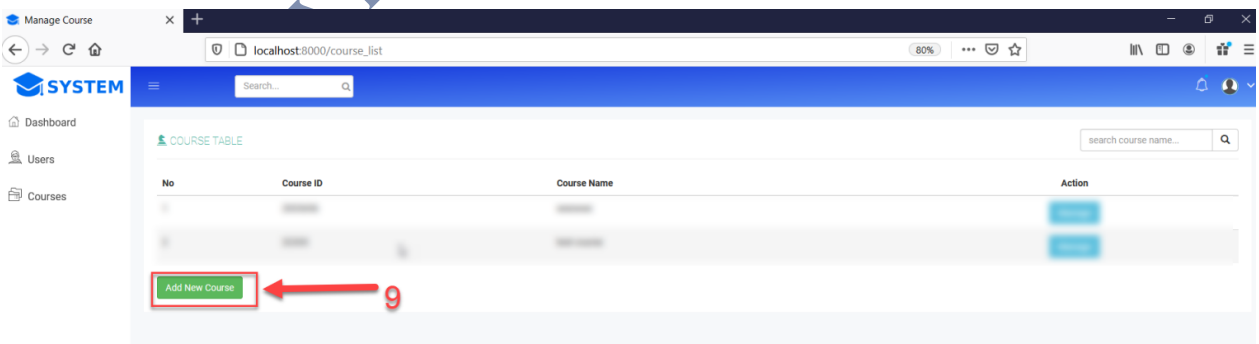

Figure 54 : User manual – Add course 2

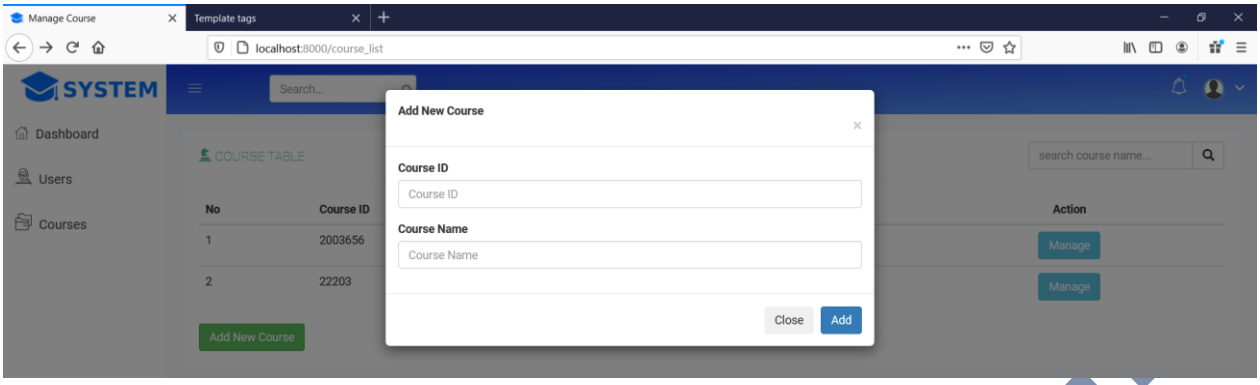

Figure 55 : User manual – Add course 3

10) To add a new section of a previously created course click on mange beside the desired course details

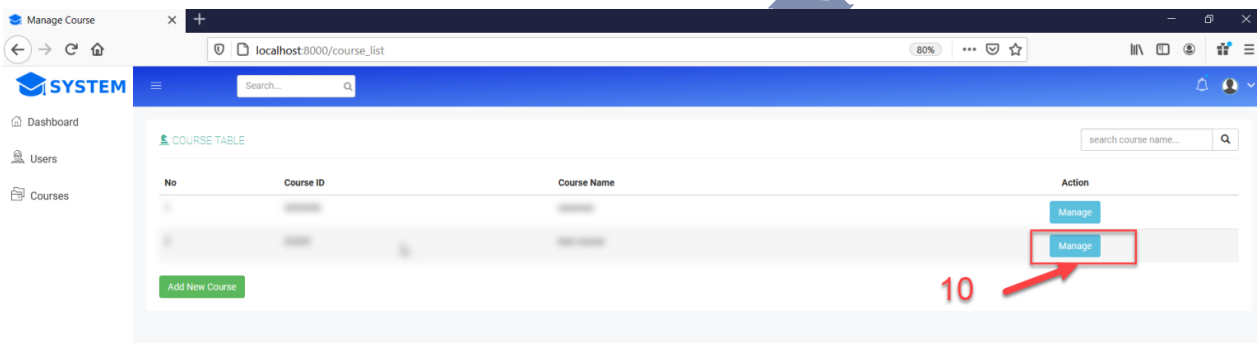

Figure 56 : User manual – Add section 1

Then to add a new section click on add new section button and fill the required information and click add button

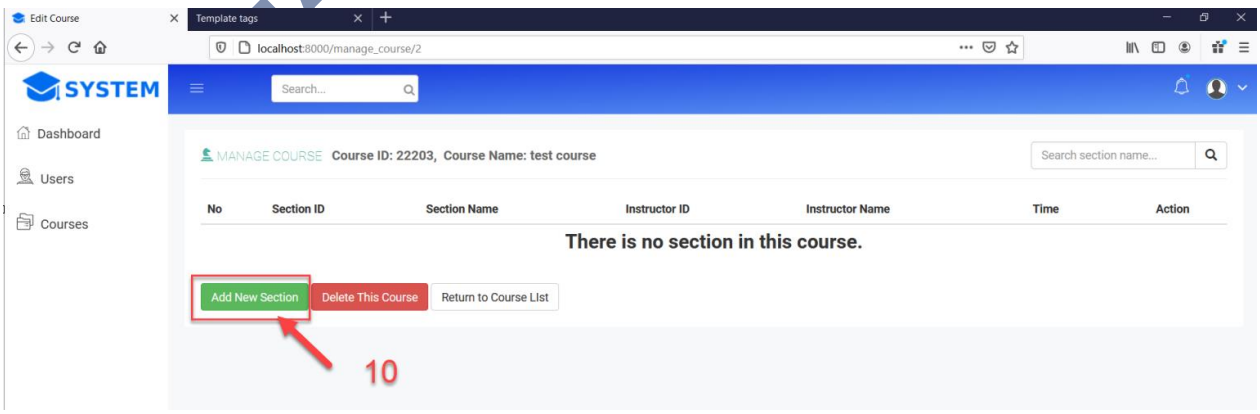

Figure 57 : User manual – Add section 2

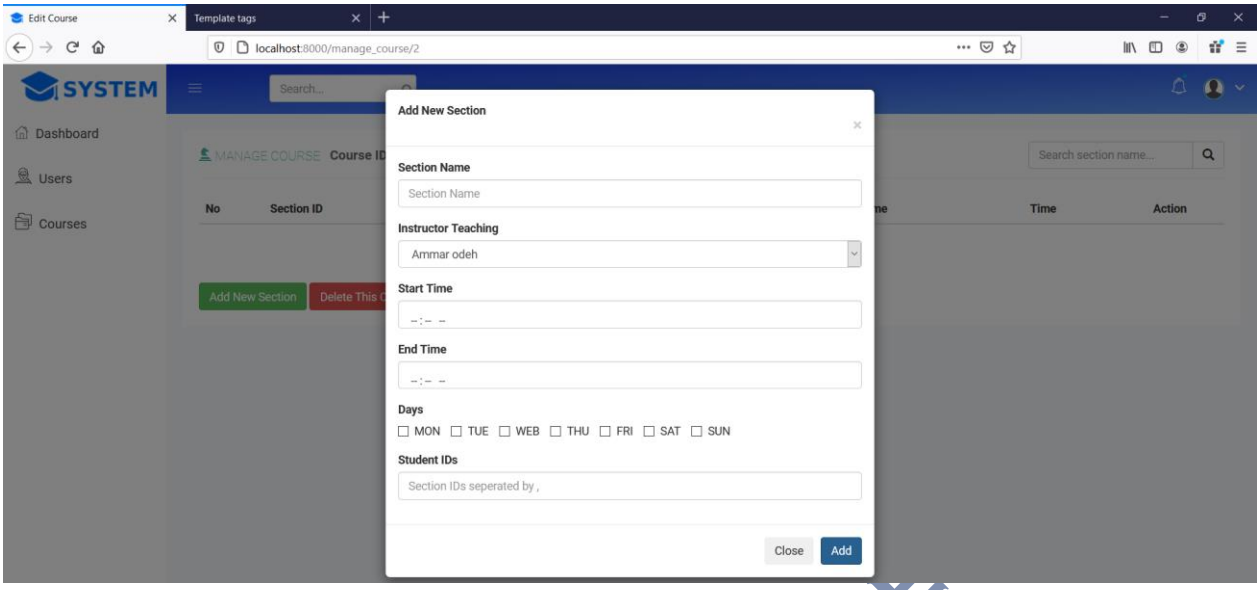

Figure 58 : User manual – Add section 3

11) For user to take the attendance of students using facial recognition the user logged in must be an instructor, from the dashboard the user clicks on the sections tab on the sidebar and selects the section and clicks take attendance button.

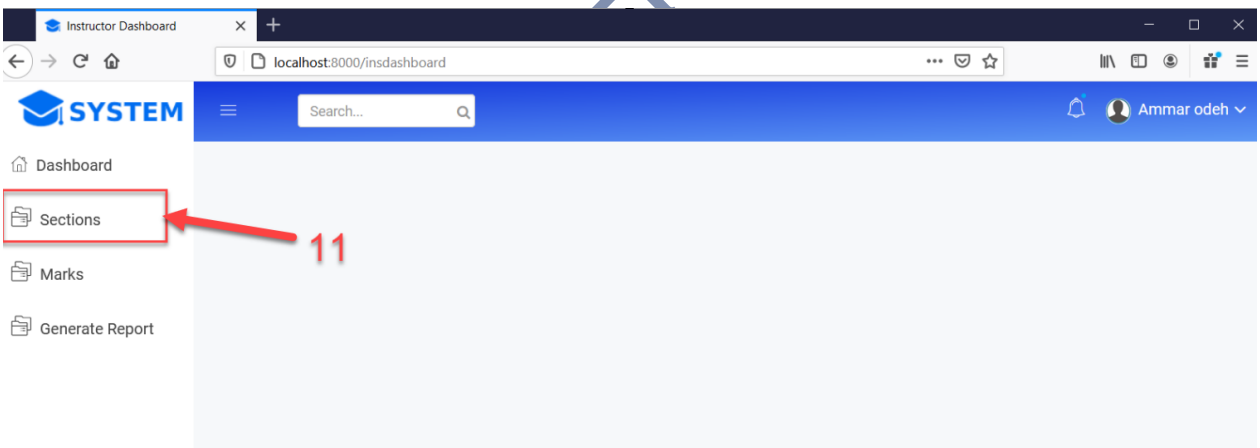

Figure 59 : User manual – Take attendance 1

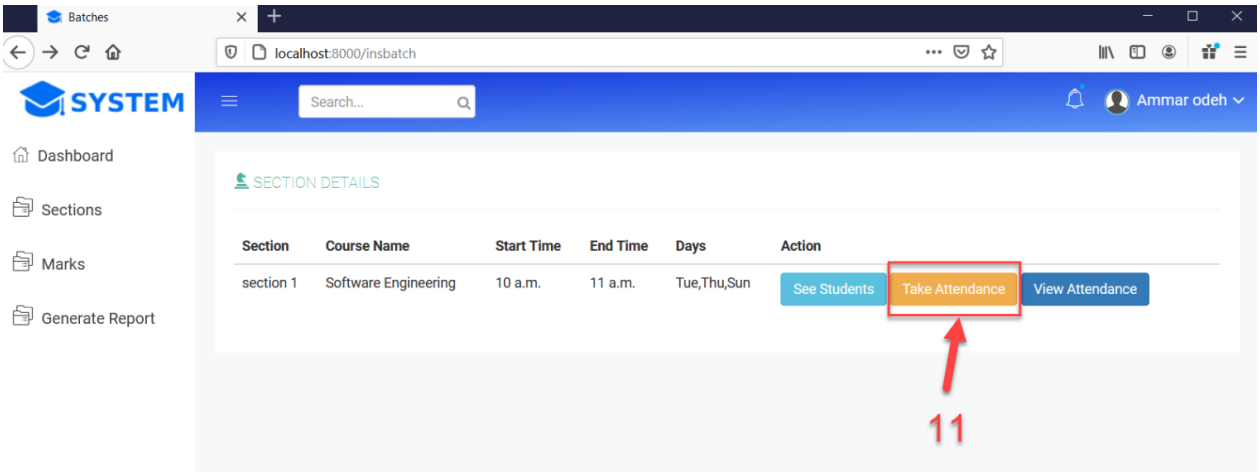

Figure 60 : User manual – Take attendance 2

To take attendance press Take Attendance button and after all the students are scanned press "S" on the keyboard then to view the attendance go back to section details page and press View Attendance button

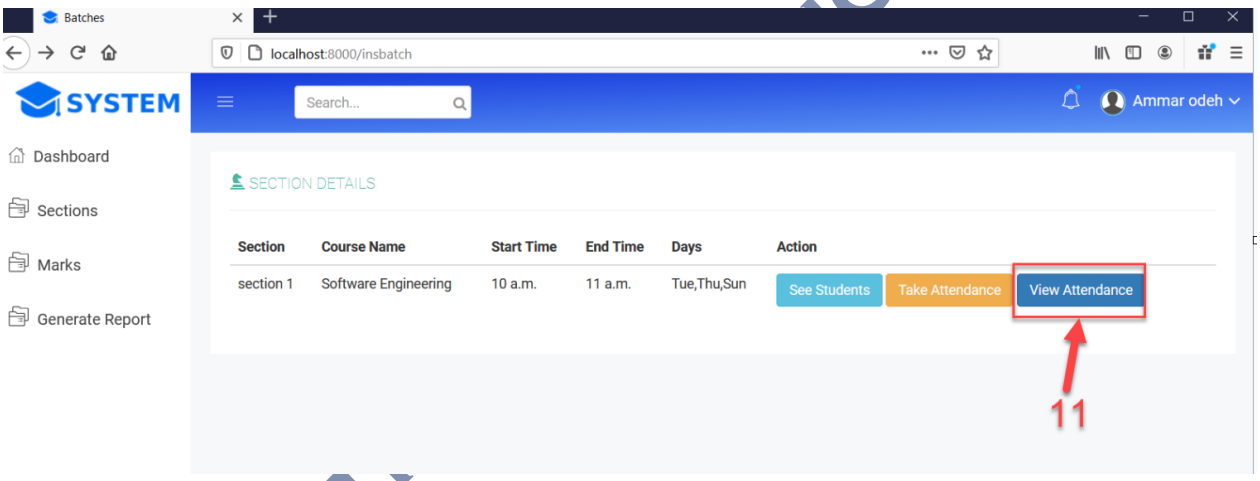

Figure 61 : User manual - View attendance 1

AUTOMATIC ATTENDANCE SY

Then select the desired date and click view attendance button

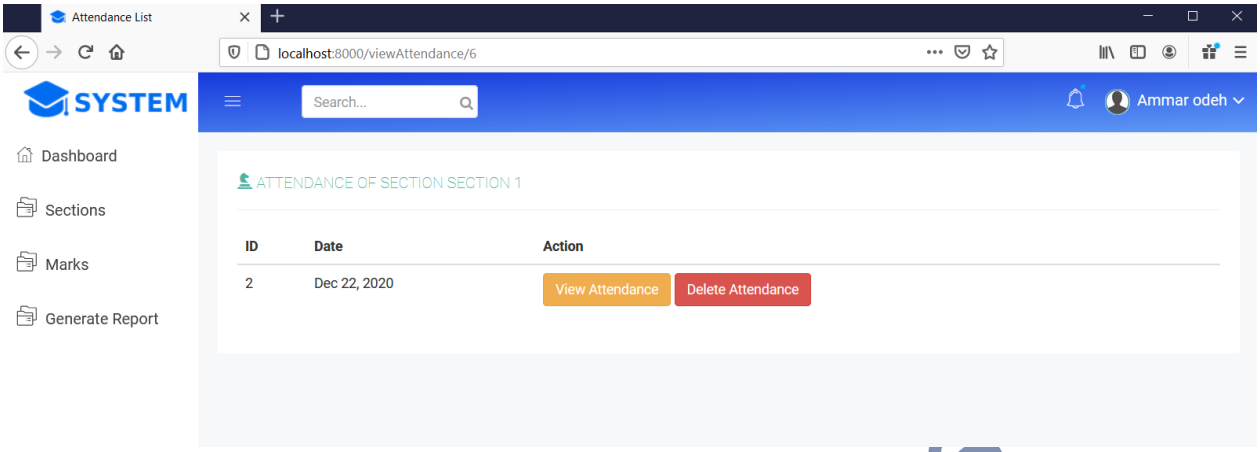

Figure 62 : User manual – View attendance 2

12) To enter grades for students, while logged in as an instructor click on Marks tab from the sidebar and click on List Marks button to the right of the desired section then enter the grades for each student

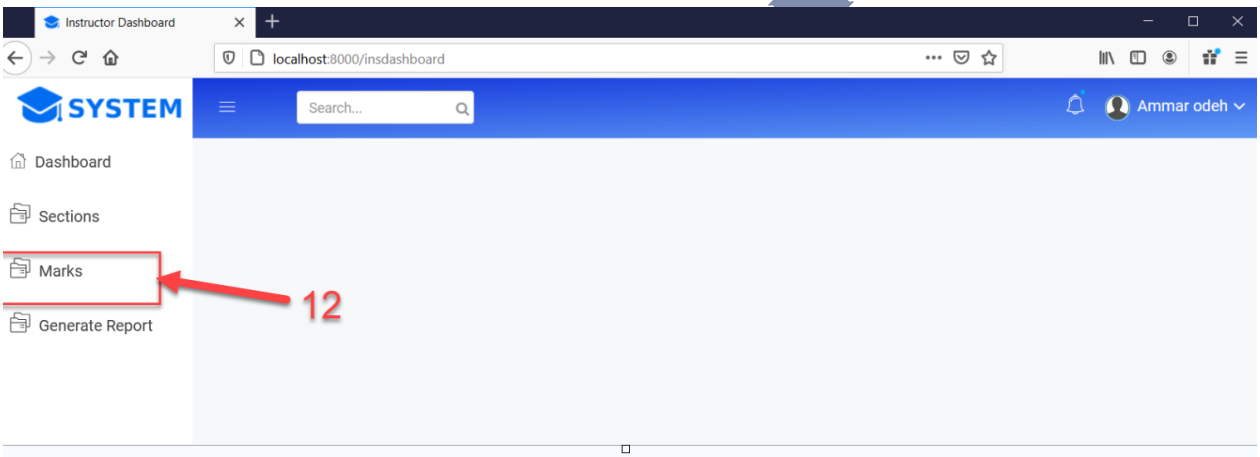

Figure 63 : User manual – Adding grades 1

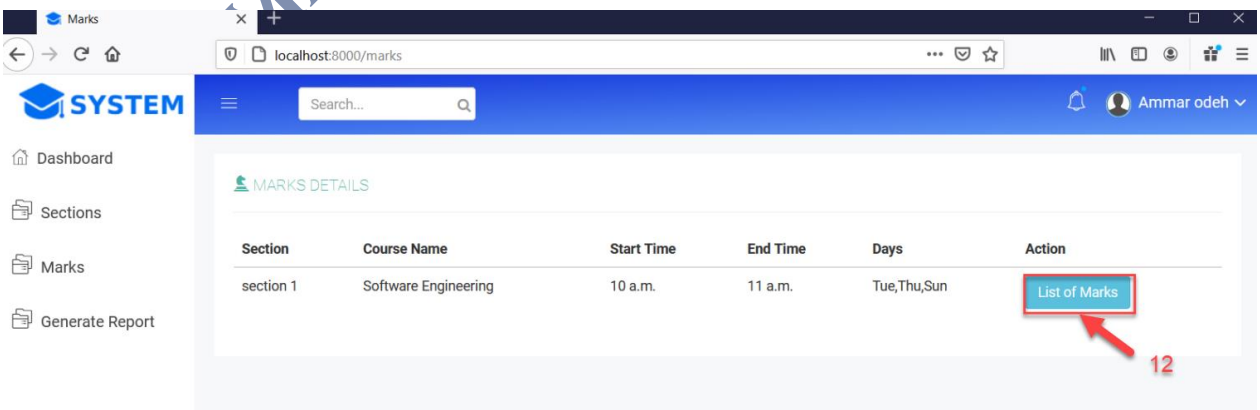

Figure 64 : User manual – Adding grades 2

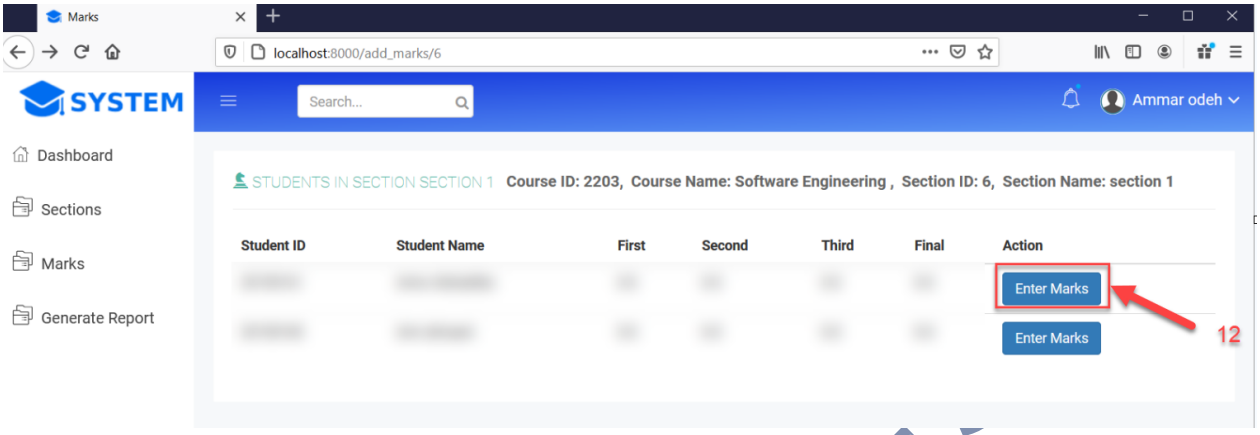

Figure 65 : User manual – Adding grades 3

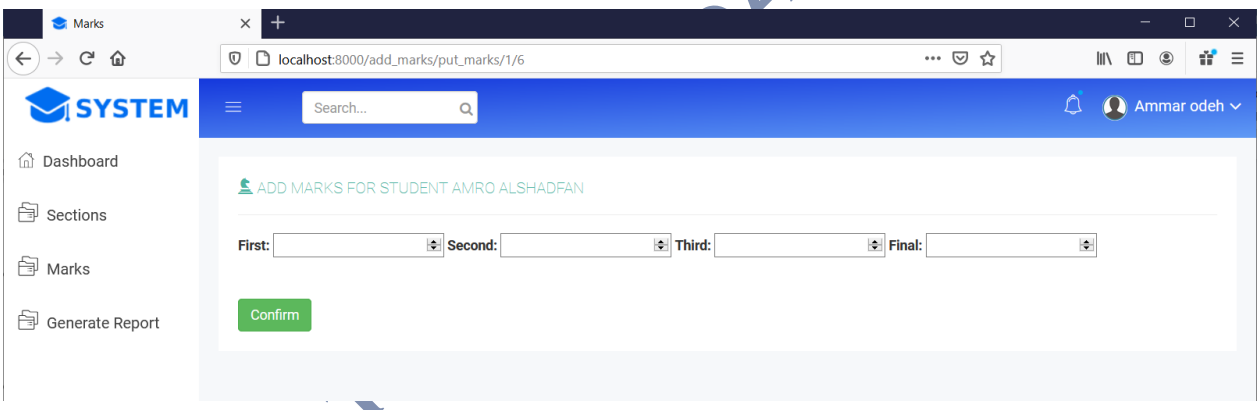

Figure 66 :User manual – Adding grades 4

AUTOMA ANCE

Appendix B
Document Changes

The following table shows the changes done on the documentation submitted on project 1 and the reason behind each change.

Table 24:Document changes

Change	The reason behind the change
Moved the implementation of the generate statistics (FR9) function to a future plan	Because of the tight time schedule and the complexity of implementing such function that requires high machine learning experience.
Proposed report designs where not implemented	Due to the moving of the generate statistics (FR9) to a future plan
Change in the proposed ER, Use Case and Class diagrams	In order to fit the requirements of implementation and different obstacles during the programing phase
Added Implementation, Testing and Future plans chapters	To fulfill the requirements of graduation project

AUTOMATIC ATTENDANCE SYSTEM

Appendix C

Code Documentation

Code Documentation is automatically generated by Django and can be accessed through the admin dashboard.

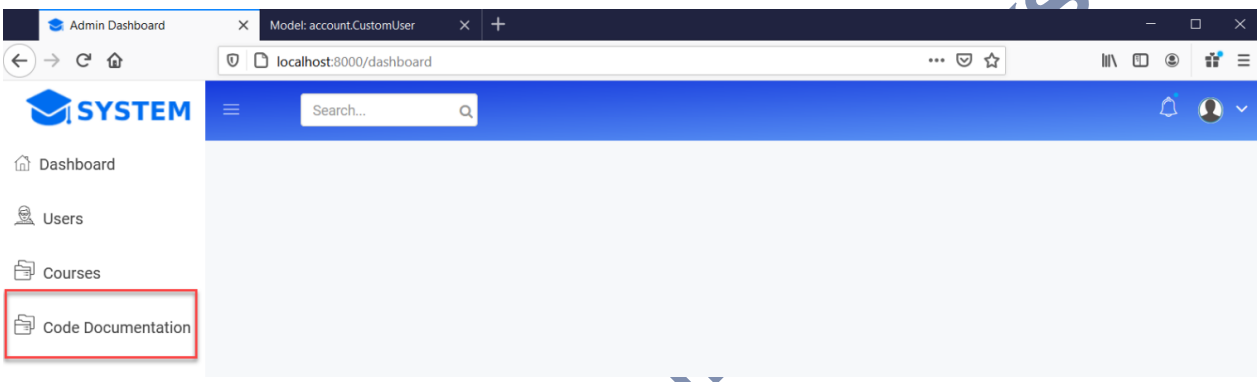

Figure 67: Code Documentation 1

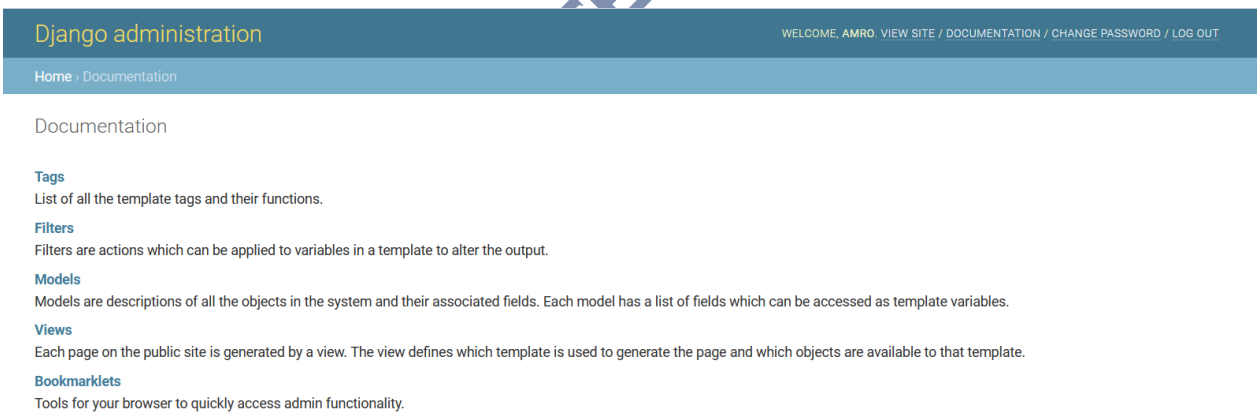

Figure 68 : Code Documentation 2

References

[1] Shoewu, O., & Idowu, O. A. (2012). Development of attendance management system using biometrics. (Shoewu) The Pacific Journal of Science and Technology, 13(1), 300-307.

[2] Chiagozie, O. G., & Nwaji, O. G. (2012). Radio frequency identification (RFID) based attendance system with automatic door unit. Academic Research International, 2(2), 168.

[3] Arulogun, O. T., Olatunbosun, A., Fakolujo, O. A., & Olaniyi, O. M. (2013). RFID-based students attendance management system. International Journal of Scientific & Engineering Research, 4(2), 1-9.

[4] Kadry, S., & Smaili, M. (2013). Wireless attendance management system based on iris recognition. Scientific Research and essays, 5(12), 1428-1435.

[5] Masalha, F., & Hirzallah, N. (2014). A Students Attendance System Using QR Code. International Journal of Advanced Computer Science and Applications, 5.

[6] Chakraborty, P., Muzammel, C. S., Khatun, M., Islam, S. F., & Rahman, S. Automatic Student Attendance System using Face Recognition.

[7] Joseph, J., & Zacharia, K. (2013). Automatic Attendance Management System Using Face Recognition.

[8] "Free Cloud Storage for Personal Use. August 2020." Google Drive, Google, www.google.com/drive/

[9] "Online Diagram Software; Visual Solution." Lucidchart, July 2020, http://www.lucidchart.com/

[10] "Diagram Software." StarUML, July 2020, http://www.staruml.io/

[11] "Online Gantt Chart Software for Project Planning." GanttPRO, June 2020, www.ganttpro.com/

[12] "A Modern, Open Source Code Editor That Understands Web Design." Brackets, Aug. 2020, ww.brackets.io/

AUTOMATIC ATTENDANCE SYSTEM