A PBL-II REPORT

ON

“Face Recognition for Mess Entry”

A PBL-II report submitted in partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science & Engineering

BACHELOR OF TECHNOLOGY COMPUTER SCIENCE & ENGINEERING

Submitted By

Ishaan Bhela	- 22070122083
Harsh Agrawal	- 22070122073
Garv Kalra	- 22070122066
Hrithik Rayapati	- 22070122082

UNDER THE GUIDANCE OF

Prof. Rahul Joshi

॥वसुधैव कुटुम्बकम्॥

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Symbiosis Institute of Technology, Pune

Symbiosis International (Deemed University)

Face Recognition for Mess Entry

A PBL-II project report submitted in partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science & Engineering

By

Ishaan Bhela	**(22070122083)**
Harsh Agrawal	**(22070122073)**
Garv Kalra	**(22070122066)**
Hrithik Rayapati	**(22070122082)**

Under the guidance of

Prof. Rahul Joshi

॥वसुधैव कुटुम्बकम्॥

CERTIFICATE

This is to certify that the PBL-I Project work entitled "**Face Recognition For Mess Entry**" is carried out by **Ishaan Bhela,** in partial fulfillment for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering**, Symbiosis Institute of Technology Pune, Symbiosis International (Deemed University) Pune, India during the academic year 2023-2024.

--------------	--------------
Dr. Sudanshu Gonge	Dr. Rahul Joshi

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Symbiosis Institute of Technology, Pune

Symbiosis International (Deemed University)

ABSTRACT

The mess entry process has been revolutionized by the Automated Mess Entry Authorization System, powered by advanced facial recognition technology harnessed by AWS Rekognition. This new system enables approved persons to access the mess facility through facial recognition, that is, their unique facial features are used to verify one's identity. In this background, the administrator will play a pivotal role in terms of this streamlined process: images of authorized persons shall be uploaded to the DynamoDB table using their respective identities, and this would be facilitated by AWS Cognito in terms of ease of user authentication.

Upon capture, the system verifies the identity of the person through highly sophisticated facial recognition algorithms, hence giving them easy access once the person is registered with the system. Otherwise, those identified as unknown are asked to meet the requirements for payment before gaining entry.

This transformative project leverages Python, AWS Rekognition, AWS Cognito, and DynamoDB technologies to integrate these technologies to deliver efficient and secure solutions for entry messaging. Placing automation, accuracy, and user experience at the center of its design, the Automated Mess Entry Authorization System sets a new standard for modernizing entry processes in educational institutions.

Keywords: Automated Mess Entry, Facial Recognition Technology, AWS Rekognition, DynamoDB, AWS Cognito, Efficiency

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AWS: Amazon Web Services

AWS Rekognition: Amazon Rekognition

AWS Cognito: Amazon Cognito

DynamoDB: Amazon DynamoDB

IEEE: Institute of Electrical and Electronics Engineers

CHAPTER 1

Introduction

1.1 Introduction

Manual verification techniques are still used in educational institutions, which creates a problem when a vast number of students are seeking entry into the mess. This ends up causing ignorance, inefficiency, and students entering the mess for free. To address this, we propose s solution, "Automated Mess Entry Authorization System". This is a step to address the issues associated with the previous, aging manual verification system. We use AWS Rekognition's power to detect and recognize human faces, to speed up the mess-entering process. This system is designed to improve efficiency, correctness, and security in mess entry through an automated process of authentication.

1.2 Problem Statement

The existing system of manually verifying entrances of mess has many drawbacks—both subjective and laborious procedures, vulnerability to human error, and inefficiency and insufficiency in handling huge numbers of disruptive users.
However, the more pungent problem is the slacking behavior of the staff members, especially during lunch hours when the students have queues and a high rush to take tests. This negligence often allows students who are without a mess card to enter the gates without payment of fees, thus realizing a great deficit for the educational institution. The unpaid entry is further compounded by the lack of a real-time mechanism for authentication, and this makes the amount of entry increase. Thus, a need to provide a modern solution to the above problems to optimize messy data entry for efficiency, correctness, and safety.

1.3 Objectives

1. Design and Implementation of the Admin Module Design a user-friendly admin module for effective control over student information. An authorized students database can be maintained with the help of administrators, who can submit pictures of the users and enter their names in the Authorized Students Database.
2. Integrating AWS Rekognition: Integrating AWS Rekognition for granting or denying entry based on face recognition. The algorithms and standards in facial recognition calls for high accuracy and efficiency in their integration into the system.
3. Design a stable architecture, but most of all that can handle a high number of students. Scalable architecture that can support future extension for the model.
4. Total Testing: Test the system for both reliability and the accuracy of facial recognition. This section tests the ability of the system to detect faces accurately under different lighting conditions, different facial expressions, and different viewing angles.

CHAPTER 2

Literature review

2.1 Background

Manual verification techniques are used in conventional mess entry management practices for educational institutions. Normally, this old practice employs manual checklists or even physical mess cards to permit access to mess to a student. However, this method has several problems.

1. Inefficiencies: There are long delays and long lines during peak hours due to the time-consuming long manual verification process.
2. Human fallacies: Manual verification will fail because staff members will often miss that a mess card is invalid or expired. This is a traditional risk of human fallacies in manual verification.
3. Volume Management Challenges: This is a lot of work because updating and maintaining manual databases turn out to be cumbersome, especially during voluminous numbers of mess users.
4. The Absence of Real-time Authentication: This makes the problem more acute, especially when unauthorized users are allowed access because of negligence or inadequate verification.

There is an instant need to seek and implement alternative practices that take advantage of technological developments to replace manual verification techniques. These problems are present because educational institutions are striving to streamline their operations to improve efficiency. This is why the use of facial recognition technology in automated systems has emerged as a promising substitute to replace manual verification techniques. These systems should bolster security, increase efficiency with mess entry, and increase accuracy by automating authentication.

2.2 Summary of literature review and research gap

The literature analysis highlights the advantages and disadvantages of current methods, offering insightful information about the state of mess entry management.

1. **Advantages of Conventional Verification Techniques:**
    a. Some traditional methods of the authentication processes have been successful in controlling mess entrance, particularly in smaller organizations of countable users.
    b. They have provided employees and students with a reliable and straightforward process.
    c. Even though these are very common, they too have some faults, which include long queues and waiting in the crowded areas, which are very unhealthy to both the staff and the students.
    d. Human error is prone to error, which may lead to unauthorized access and eventually loss in the business.

e. These problems are compounded by the complexity of the management of a large number of mess users and calls for an extraordinary labor and resource involvement.

2. **Automation's Rise to Power:**

   a. The literature study further emphasizes how automated solutions, particularly those that incorporate facial recognition technology, are coming up as valid replacements for traditional methods.
   b. These automated methods ensure greater security and accuracy. They speed up the process of entering the mess and is also great in not letting unauthorized persons go in without paying the mess fee.
   c. Case studies and research findings have shown that facial recognition technology works in various applications, from identity verification to access control.

Feature	Manual Verification	Barcode/RFID	Biometrics	Facial Recognition
Accuracy	Low (human error)	Moderate	High	High
Convenience	Low (manual process)	Low (cards/tags)	Moderate(fingerprints/iris scans)	High (contactless)
Scalability	Limited	Moderate	Moderate	High
Security	Low (potential for unauthorized access)	Moderate (card cloning/misplacement)	High (unique biometrics)	High (face is unique, no physical contact)
Privacy	High (no data collection)	Moderate (anonymous card data)	Low (biometric data collection)	Moderate (no sensitive data used)
User Acceptance	High (familiar)	Moderate (cards/tags)	Low (privacy concerns)	High (convenient and familiar)
Cost	Low (minimal setup)	Moderate (cards/tags, readers)	High (scanners, infrastructure)	Moderate (camera, cloud services)

Table 2.1 Comparing Different Solutions

Research Gap: Even though an automated mess entry management system is on the rise, there is still little information available in the literature regarding how to use AWS Rekognition to implement these systems. Although facial recognition technology has already been developed and implemented in various applications, there is less information about how such technology can be used within the scope of mess entry management using AWS Rekognition. This study proposes an automated mess entry authorization system based on AWS Rekognition and attempts to fill this information gap and contribute to the existing body of information on automated authentication systems and mess entry management.

CHAPTER 3

Software Requirements Specification

3.1 Software Tool Platform/ Tools/Framework Used

To make the Automated Mess Entry Authorization System effective, and reliable and free from manual verification methods, numerous tools and platforms are used at the time of its design and development. Essential elements include:

1. Python Programming Language: The main programming language used to build the system is Python. Because of its ease of use, flexibility, and an libraries which facilitate us to make an easy connection with AWS resources. it's ideal for quick development and AWS service integration.
2. AWS Services: DynamoDB, AWS Rekognition, and AWS Cognito: In view of the sophistication of the facial recognition capabilities of AWS Rekognition, the system can recognize and authenticate users who want to access by using their facial attributes.
3. AWS Cognito: An authentication tool for administrators to login into the website. The administrators can manage the mess student database only by first logging into the website. The user identity and passwords are stored safely inside AWS Cognito.
4. DynamoDB: The database used to store user information and facial recognition output. Its scalability, generality, and low latency performance make it ideal for real-time data storage and retrieval.
5. Flask: A is the web framework, for the admin module. The admin module is implemented using Flask, a lightweight and versatile Python web framework.
6. HTML for front-end: The website's and the admin module's front-end are developed using HTML Hyper-Text-Markup-Language. Web pages are structured and ordered according to HTML. Webpages like Login page and the upload image page is made using HTML and is user friendly.

The Automated Mess Entry Authorization System is built to be reliable, effective, and user-friendly under crowded conditions. Providing a flawless experience for admins to manage the student database inside AWS DynamoDB. Moreover, the system's automation and real-time processing are guaranteed by using Flask, a Python Library which will manage the website flow using API calls and AWS Lambda which will handle the data flow inside the cloud.

3.2 Functional Requirements

1. User Authorization and Authentication
    a. Admin Login: For secure login interface, the system should include some sort of module for the administrators.
    b. Implement Authentication: To only allow such authorized administrators to log in and control user data, there is a need to develop authentication processes.
2. User Administration
    a. Student Addition: Submission of photos and the IDs attached to them would allow the adding of new students to the system.
    b. Deletion of Students: Administrators should also have the capability of removing or deactivating student records in the system.
    c. Uploading and Processing Images
    d. Image Upload: Administrators should have the ability to upload images of student face who are allowed to enter the mess.

    e. Image Processing: The system should be able to automatically create face IDs of uploaded student images using AWS Rekognition, after which it would be added into DynamoDB.

3. Entry Authorization and Facial Recognition
    a. Real Time Face Recognition: On coming to apply for mess entry, the system should be able to recognize his or her face in real time.
    b. Entry Authorization: Mess entry authorization should only open for duly identified authorized personnel.
    c. Unauthorized Entry Handling: Unrecognized people should be turned away and asked to pay.
4. Connectivity to AWS Services
    a. Integration of AWS Rekognition: Through this, it would be able to apply the facial recognition feature.
    b. Integration of DynamoDB: Storage and retrieval of student records and test results using face IDs for unauthorized mess entry should be using DynamoDB.
5. Use of Lambda Functions for Automation
    a. Automation Event-Driven: Lambda function must be able to manage the dataflow inside the AWS services and store the FaceID generated safely inside the DynamoDB.

3.3 Non-Functional Requirements

1. Performance
    a. Real-Time Processing: To validate entry fast, the system needs to process facial recognition tasks in real time.
    b. Scalability: To withstand increases in the number of students seeking entry in the mess and image uploads, the system should be scale itself without sacrificing performance.
2. Efficiency
    a. The system should use all of its available resources to their fullest effect. The least amount of memory and processing time should be used to process information from images.
3. Safety
    a. Authentication Security: To block the theft of the administrator's credentials and to prevent unauthorized access to the administration module, secure authentication methods will have to be applied.
4. Dependability
    a. High Availability: To minimize the downtime and ensure an uninterrupted work process even in the case of a large amount of load, the system must have a mindset of high availability.
    b. Fault Tolerance: The system should use built-in error recovery and system stability methods to withstand malfunctions and other faults.
5. Integration
    a. AWS Service Integration: The system must be seamlessly integrated with the functionalities and resources provided by AWS services (Rekognition, Cognito, and DynamoDB).
    b. Compatibility: The system must be compatible with the majority of web browsers and devices to ensure it will always function well.

Chapter 4

Methodology

4.1 System Architecture

The system architecture of Automated Mess Entry Authorization System describes a general layout, required parts, and interrelationships between AWS services and local applications that are required to accomplish the functionality that has been envisioned. This section provides an elaborate description of the system architecture, including the frontend and backend components as well as the relationship with AWS services.

System Architecture Components:
Components of System Architecture:

1. Frontend (Admin Module and Website):
    a. HTML/CSS/JavaScript: The frontend interface utilizes HTML and JavaScript to make an interactive Admin Module. This FrontEnd provides admins to upload student's faces and names so that its stored inside the database.

2. Backend Services:
    a. Python with Flask: The backend services are build using Python in Flask to render the application logic for user authentication, image processing, and the database operations.
    b. AWS Lambda: Usage of Serverless Lambda functions to automate tasks triggered by S3 events, since it entails image processing and database update.

3. Database:
    a. DynamoDB: Amazon DynamoDB is the NoSQL database that will be used to store authorized students or the students who have paid the mess fee. The system uses scalability, low latency, and easy integration with other AWS services.

4. AWS Services:
    a. AWS Rekognition: The system uses Rekognition as an integrated service for facial recognition capabilities, which involves face detection, face matching, and facial feature extraction.
    b. AWS Cognito: It is used to provide user authentication and management, securing easy and simple access control and user management in general.
    c. Amazon S3 (Simple Storage Service): Used to store and manage uploaded images and to trigger Lambda functions to process the uploaded images automatically.

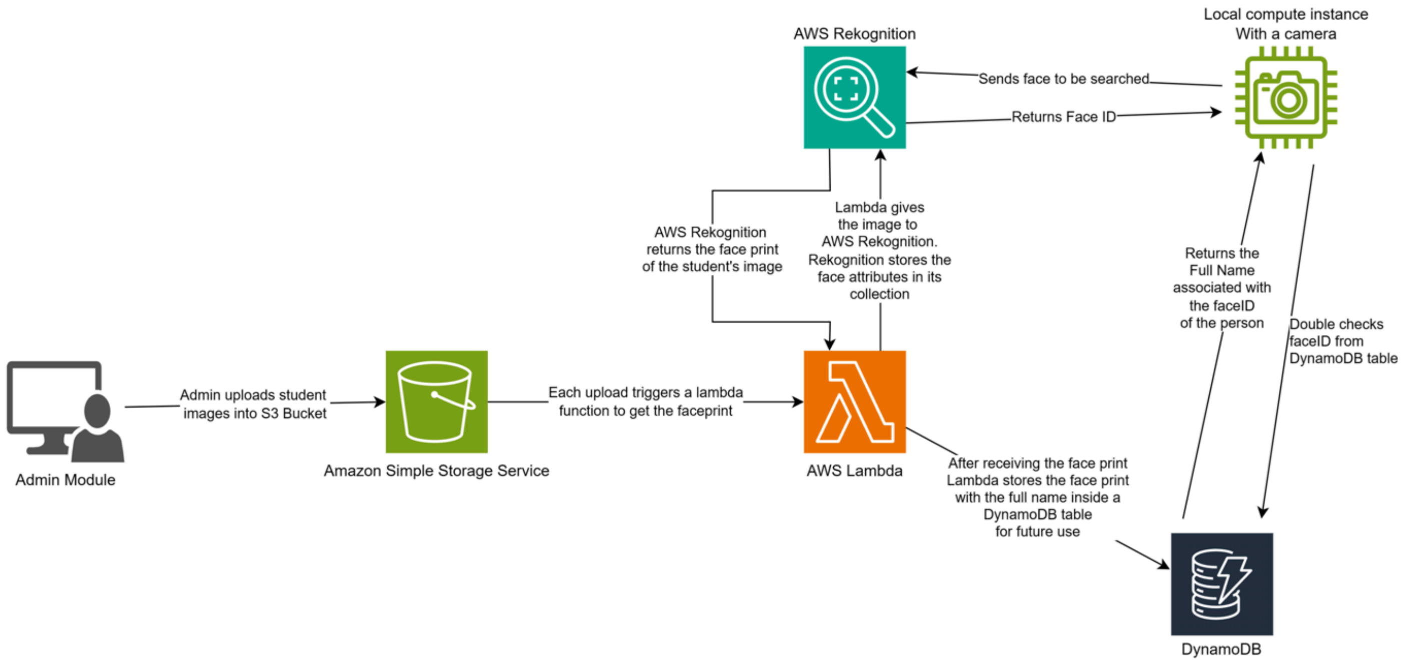

AWS Rekognition
Local compute instance With a camera
Sends face to be searched
Returns Face ID
AWS Rekognition returns the face print of the student's image
Lambda gives the image to AWS Rekognition. Rekognition stores the face attributes in its collection
Returns the Full Name associated with the faceID of the person
Double checks faceID from DynamoDB table
Admin uploads student images into S3 Bucket
Each upload triggers a lambda function to get the faceprint
Admin Module
Amazon Simple Storage Service
AWS Lambda
After receiving the face print Lambda stores the face print with the full name inside a DynamoDB table for future use
DynamoDB

Figure 4.1 System Architecture

4.2 Data Flow

Important Elements of a Data Flow Diagram:

1. Control Panel (Frontend)
    a. Functionality: Provides a platform where administrators can interact with the system, offer photos, and manage student data.
    b. Examples of data inputs: users' actions, such as adding students, uploading photos, and configuring settings of the system.
    c. Data Outputs: Output from the backend system, such as error messages or success messages will be displayed inside the frontend.
2. Backend Python (Flask Application)
    a. Functionality: We will be using Flask to control every action inside the frontend. The Flask application has all the routings and whenever a button is clicked in the frontend, it will go through the Flask server to see what action is to be performed next.
    b. Data Inputs: When ever an Admin uploads an image of student and the name is provided and clicks on the submit button, the Flask API cones into action. It takes all the data inputs and uploads into the database accordingly.
3. Functionality of AWS Services (Rekognition, Cognito, DynamoDB, Lambda):
    a. These services are used for face recognition technology, admin module security, database for students and flow of data inside the cloud.
    b. Data Inputs: Data gathered from the backend application, that is the image of student and name will be collected by AWS Lambda and will go for further processing.
    c. Data outputs: After the image is processed in AWS Rekognition it updates to the data to the AWS DynamoDB table. and gives a respond to the the backend

application, such as new student records and facial recognition outputs; facial recognition outputs (face IDs), and authentication outputs.

4. Database (DynamoDB) Functionality:
    a. Data items such as user name, and face IDs retrieved by facial recognition are stored in DynamoDB tables.
    b. Data inputs: updated data from the backend application, such as new student records and face IDs retrieved by facial recognition and the Face ID returned by Rekognition is put into DynamoDB asa new record.
    c. Data Outputs: Whenever a person wants to enter the mess, the face captured by the webcam is sent to Rekognition and the ID generated by Rekognition is checked by the backend inside the DynamoDB

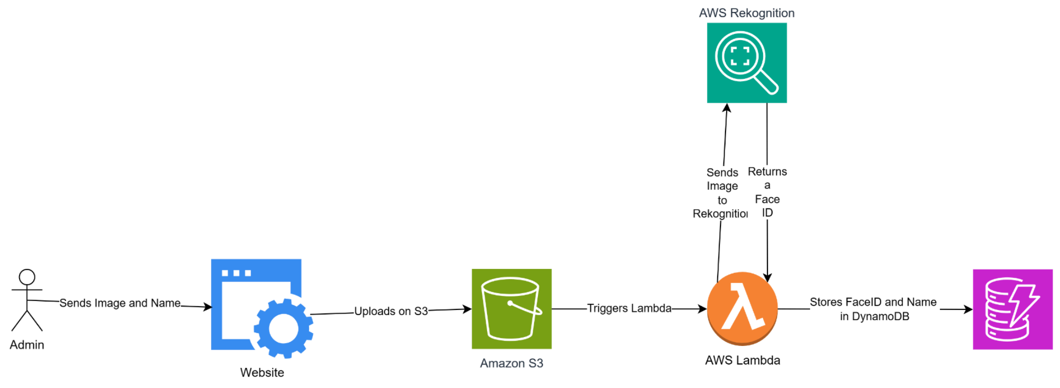


Figure 4.2 Data Flow Diagram

4.3 Implementation Details

Significant Components of Implementation:

1. Frontend Development:
    a. The technologies used in the admin module and website are HTML and JavaScript.
    b. The frontend interface, including forms for user input and picture uploads, is organized using HTML.
    c. The frontend components are styled using CSS, ensuring aesthetic beauty and intuitive user interface.
    d. JavaScript manages dynamic content changes, on the front end.
2. Backend Development with Flask and Python:
    a. Tools: Flask framework, Python
    b. Logic related to user authentication, picture uploading, and website flow is implemented in Python.
    c. Flask is a lightweight Python web framework used to handle HTTP requests from the front end and develop RESTful APIs.
3. AWS Service Integration:
    a. Tools: AWS SDK (Boto3) Description: The AWS SDK for Python (Boto3) is used to integrate AWS services like Lambda, Rekognition, Cognito, and DynamoDB.

    b. Boto3 makes it possible to integrate various AWS features such as data storage, user authentication, facial recognition, and event-driven automation by making the connectivity to AWS services available.

4. DynamoDB Database Management:
    a. Tools: Amazon DynamoDB table.
    b. Description: Face IDs obtained from AWS Rekognition and student name obtained from the website are stored in DynamoDB tables, a NoSQL Database with the help of AWS Lambda.
    c. Based on the system requirements, tables and indexes are built to ensure a smooth insertion and retrieval of data.
5. AWS Lambda Functions (Serverless Computing)
    a. Description: AWS S3 events, such as picture uploads, triggers the AWS Lambda which automates tasks like adding FaceID into the database.
    b. Lambda Function ensures smooth flow of data inside the cloud.

Code Algorithms and Techniques:

1. Modular Design: To promote maintainability and scalability, the system is modularly constructed, encapsulating functionalities into distinct modules, such as image processing and authentication.
2. Version Control: Git is used to manage the versions of the project, ensuring code sharing, collaborative development, and tracking changes.
3. Testing and Debugging: Loading methods are put in place to track errors and debug, and unit testing is carried out to validate the components.

4.4 Testing Strategies

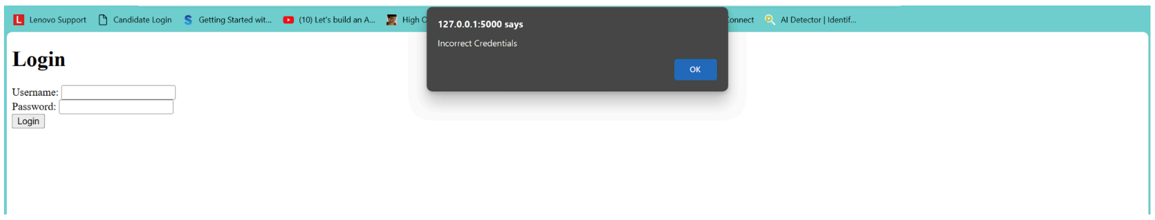
Test-Related Activities:

1. Website Front-end: Testing Objectives: Validating the functionality of uploading student images and the appearance of a successful upload notification. Also handling errors when a file of non-image format is added like.gif etc.
    a. Steps: Open the website.
    b. Open the uploading section where the pictures are uploaded.
    c. Choose a picture file, then upload it.
    d. Check if the upload was performed without an error.
    e. Check if a success message appears when the work is done.
2. Back-end Model Testing Objectives: Validate face detection and recognition functionality
    a. Steps: To conduct the test use a test image or a webcam to imitate taking an image.
    b. Present the face recognition model with the picture taken.
    c. Check if the model can detect any face in the picture shown.
    d. Check if appropriate, that the identified face is the face of a student that is registered.

Test Scenarios:

1. Pictures Upload to Website:
    a. Ensure all picture file extensions like JPG and PNG are uploaded without any error.
    b. Performance of upload process for large as well as small picture files.
    c. Ensure that the invalid or unsupported file formats are handled through error handling.
2. Back-end Model Face Detection:
    a. Ensure that the targeted face is detected correctly.
    b. The model responds quickly to various image sizes and quality settings.
    c. Ensure the face detection accuracy of the model through different lighting as well as angle settings.

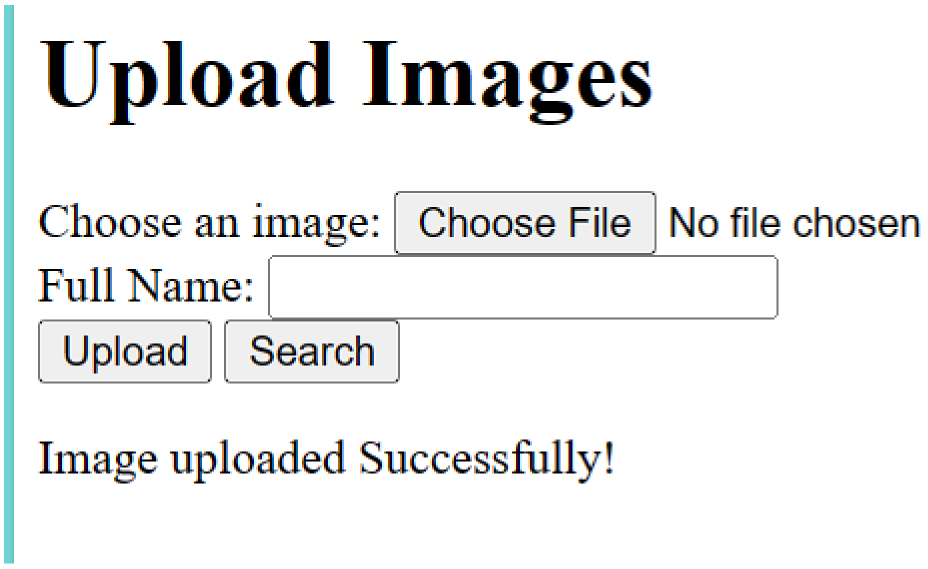
Website Testing



127.0.0.1:5000 says

Incorrect Credentials

——-------------------------------------Figure 4.3 Login Error——---

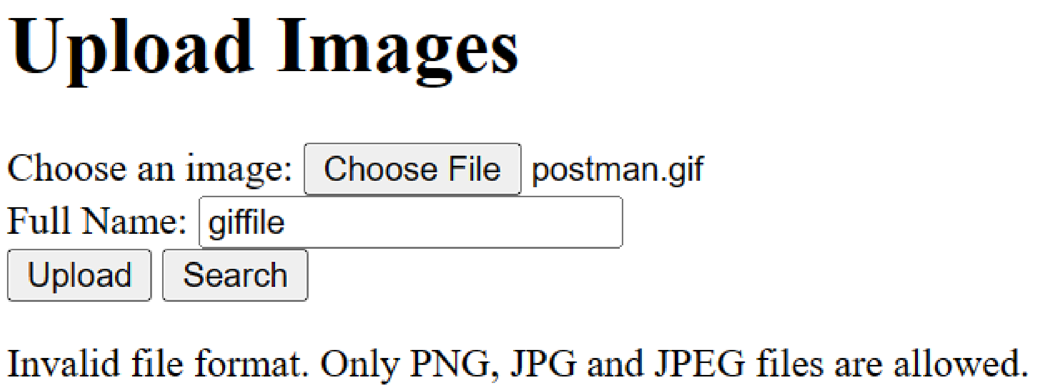

Upload Images

Choose an image: No file chosen

Full Name:

Image uploaded Successfully!

——-----------------------------Figure 4.4 Image Uploaded Successfully——---------------------------



Upload Images

Choose an image: Choose File postman.gif

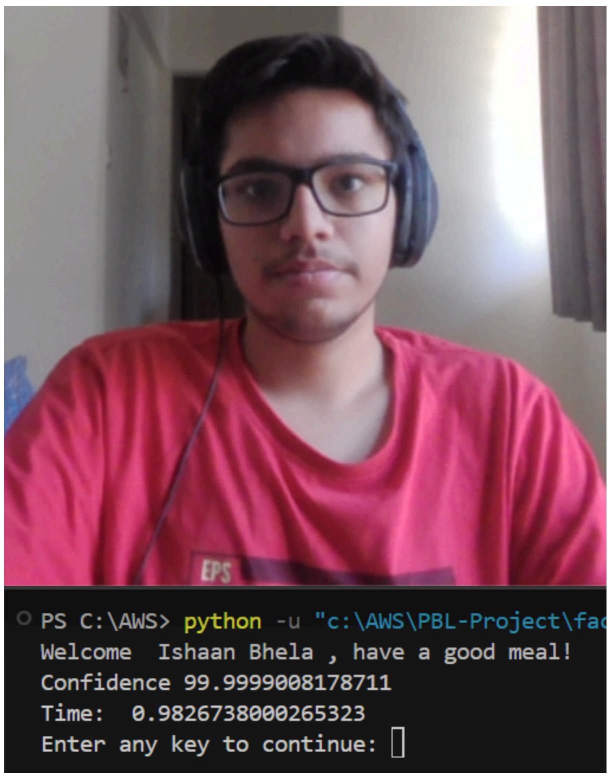
Full Name: giffile

Upload Search

Invalid file format. Only PNG, JPG and JPEG files are allowed.

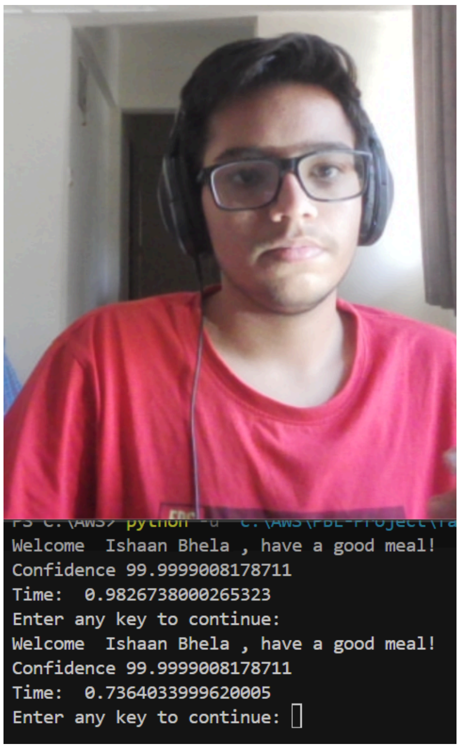
——-------------------------------Figure 4.5 Error based on File Extension——---------------------------

Model Testing



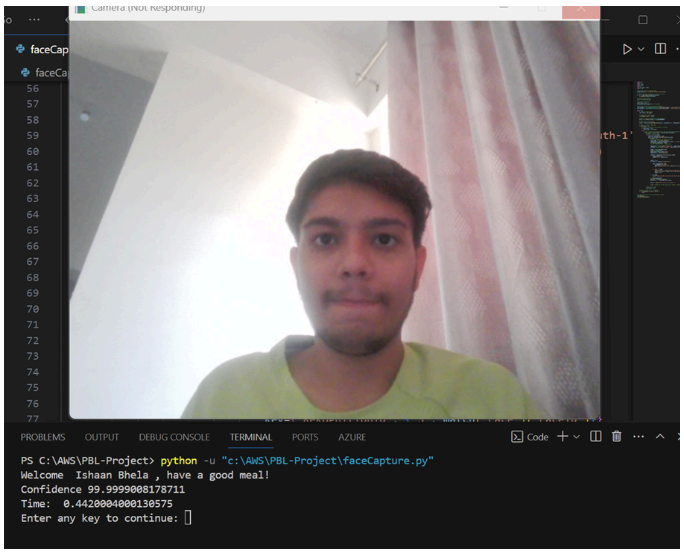
```
PS C:\AWS> python -u "c:\AWS\PBL-Project\fac
Welcome  Ishaan Bhela , have a good meal!
Confidence 99.9999008178711
Time:  0.9826738000265323
Enter any key to continue: 
```

Figure 4.6 Model Output (Normal)



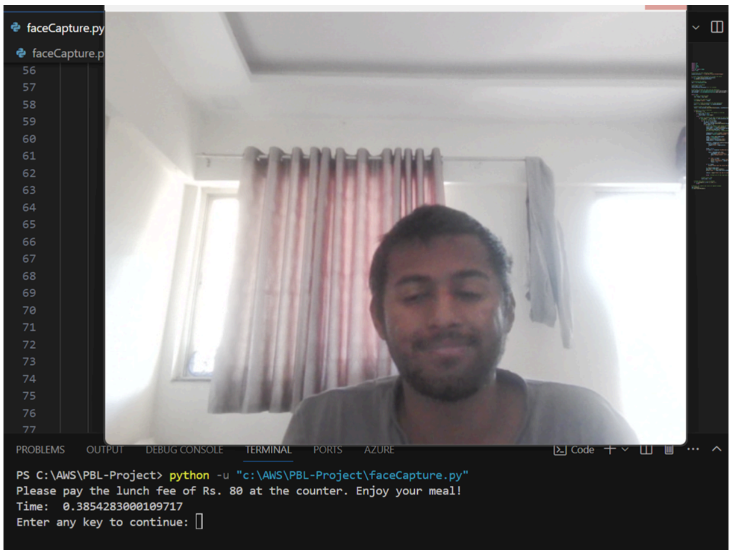
```
PS C:\AWS> python -u "c:\AWS\PBL-Project\fa
Welcome  Ishaan Bhela , have a good meal!
Confidence 99.9999008178711
Time:  0.9826738000265323
Enter any key to continue:
Welcome  Ishaan Bhela , have a good meal!
Confidence 99.9999008178711
Time:  0.7364033999620005
Enter any key to continue:
```

Figure 4.7 Model Output (Light Conditions)



```
PS C:\AWS\PBL-Project> python -u "c:\AWS\PBL-Project\faceCapture.py"
Welcome  Ishaan Bhela , have a good meal!
Confidence 99.9999008178711
Time:  0.4420004000130575
Enter any key to continue:
```

Figure 4.8 Model Output (Without specs)



```
PS C:\AWS\PBL-Project> python -u "c:\AWS\PBL-Project\faceCapture.py"
Please pay the lunch fee of Rs. 80 at the counter. Enjoy your meal!
Time:  0.3854283000109717
Enter any key to continue:
```

Figure 4.9 Unrecognized Person

Chapter 5

Results and Discussion

5.1 Summary of Results and Discussion

The Automated Mess Entry Authorization System has produced phenomenal results from its implementation and testing, highlighting the system's features and outlining important observations and conclusions.
A Summary of the results.

1. With the use of HTML and JavaScript, the website provides the feature to post images.
2. At the area of Mess Entry, the camera will automatically take a picture of the student’s face will send it to AWS Rekognition and DynamoDB to recognize the person.
3. The facial recognition model shows great accuracy in the detection of faces when associated with AWS Rekognition.
4. Test scenarios validate the model's capability to recognize faces in photos taken with time webcam.
5. The solution smoothly integrates database and image processing with AWS services. This includes Lambda, DynamoDB, and Recognition
6. Safe user authentication and access control to the system is assured by AWS Cognito.

Discussion:

1. Usability & User Experience:
    a. The front end on the website makes it easy for administrators to post photos and keep records of students.
    b. User testing feedback focused on ease of follow instruction and user-friendly design.
2. Performance & Scalability:
    a. According to performance tests, the system can handle facial recognition work and image upload appropriately.
    b. Scalability factors are measured to ensure a stable system when workloads fluctuate, including AWS infrastructure and service limitations. AWS infrastructure are up 24/7 and it can process any number of faces but one at a time. The average time to recognise a person and authorize or deny entry is null and void considering other factors like, the time taken by a person in front of us to finish taking his/her food and etc.
3. Accuracy & Reliability:
    a. Facial recognition model accuracy ultimately determines reliable entry of messages.
    b. Goals include testing and validation to optimize model parameters and improve the algorithms of detection.

Chapter 6

Conclusion

6.1 Conclusion

Constituted by the development and implementation of the Automated Mess Entry Authorization System, modernizing and making the process of mess entry easy for students as well as the staff in educational institutions has now been possible. This chapter presents a summary of the project results and achievements, with special consideration to the key lessons learned and closing thoughts.

Important Conclusions:

1. Efficiency Gains:
   a. The automated facial recognition technology replaces the manual human verification which involves a mess staff to always be present at the gate to check if the student have paid the fee or not and thus saving wait times and increasing overall operational effectiveness.
   b. Facilitated student administration and real-time authentication increase supervision of mess entry for administrators.
2. Strengthened Security and Control:
   a. The solution minimizes the illegal entries and revenue loss by increasing the strength of the security measures and access control using AWS Rekognition and DynamoDB.
   b. Fortified authentication protocols increase security and accountability of the mess entry system.
3. Cost-effective solution:
   a. Reduced costs for the project, and maximized instructional value through the use of open-source technology and AWS Free Tier services.
   b. The project demonstrates that state-of-the-art technologies can be brought to reality with a limited budget.
4. Better User Experience:
   a. The automated procedures along with the intuitive website design improves the user experience for both administrators and users making a request for authorization of mess admission.
   b. This can improve user interfaces and accessibility features in future generations.

6.2 Future Scope:

1. Provide real-time update, reminders, and status of mess entry to the administrators as well as the users.
2. Advanced Analytics: Advanced analytics provides valuable information regarding user behavior, attendance records, and mess entry patterns at the different times of the day. This will help mess know when the crowd will be the most and then they will be prepared to handle the crowd before time.
3. App for mobile phones: There needs to be an application developed for mobile phones and tablets, for easy access to messages. The Application can also be used for other

things like a virtual mess card, mess fee payment poeral and checking how many days are left before someone's membership expires.

4. Attendance tracking: Facility for automating student attendance tracking and reporting for academic purposes.

References

[1] Amazon Web Services, Inc. (n.d.). AWS Documentation: Amazon Rekognition. Retrieved from https://docs.aws.amazon.com/rekognition/

[2] Amazon Web Services, Inc. (n.d.). AWS Documentation: Amazon DynamoDB. Retrieved from https://docs.aws.amazon.com/dynamodb/

[3] Amazon Web Services, Inc. (n.d.). AWS Documentation: AWS Lambda. Retrieved from https://docs.aws.amazon.com/lambda/

[4] Amazon Web Services, Inc. (n.d.). AWS Documentation: AWS Cognito. Retrieved from https://docs.aws.amazon.com/cognito/

[5] Flask Documentation. (n.d.). Flask Documentation. Retrieved from https://flask.palletsprojects.com/en/2.0.x/

[6] Mozilla Developer Network. (n.d.). HTML: Hypertext Markup Language. Retrieved from https://developer.mozilla.org/en-US/docs/Web/HTML

[7] W3Schools. (n.d.). CSS Tutorial. Retrieved from https://www.w3schools.com/css/

[8] JavaScript MDN Web Docs. (n.d.). JavaScript. Retrieved from https://developer.mozilla.org/en-US/docs/Web/JavaScript

[9] Python Software Foundation. (n.d.). Python Documentation. Retrieved from https://docs.python.org/3/

[10] IEEE Citation Reference. (n.d.). IEEE Editorial Style Manual. Retrieved from https://www.ieee.org/publications/rights/style_manual.pdf

Plagiarism Report and AIC Form

Proforma 4

Undertaking from the UG/PG student(s) while submitting his/her final dissertation to his respective institute

Ref. No. _________

I / We, the following student(s)

Sr. No.	Sequence of students names on a dissertation	Students name	Name of the Institute & Place	Email & Mobile
1.	1st author	Ishaan Bhela	Student SIT, Pune.	Ishaan.bhela.btech2022@sitpune.edu.in
2.	2nd author	Harsh Agrawal	Student SIT, Pune.	Harsh.agrawal.btech2022@sitpune.edu.in
3.	3rd author	Hrithik Rayapati	Student SIT, Pune.	Hrithik.rayapati.btech2022@sitpune.edu.in
4.	4th author	Garv Kalra	Student SIT, Pune.	Garv.kalra.btech2022@sitpune.edu.in

Note: Put additional rows in case of more number of students

hereby give an undertaking that the dissertation entitled "Face Recognition For Mess Entry" has been checked for its Similarity Index/Plagiarism through TURNITIN software tool; and that the document has been prepared by me/us and it is my/our original work and free of any plagiarism. It was found that:

1.	The Similarity Index (SI) was: *(Note: SI range: 0 to 10%; if SI is >10%, then authors cannot communicate ms; **attachment of SI report is mandatory**)*	6 %
2.	The ethical clearance for research work conducted obtained from: *(Note: Name the consent obtaining body; if 'not appliable' then write so)*	not applicable
3.	The source of funding for research was: *(Note: Name the funding agency; or write 'self' if no funding source is involved)*	self
4.	Conflict of interest: *(Note: Tick whichever is applicable)*	No
5.	The material (adopted text, tables, figures, graphs, etc.) as has been obtained from other sources, has been duly acknowledged in the manuscript: *(Note: Tick whichever is applicable)*	Yes

In case if any of the above-furnished information is found false at any point in time, then the University authorities can take action as deemed fit against all of us.

[illegible] | Harsh Agrawal | [illegible] [illegible] [illegible]

Ishaan Bhela | Harsh Agrawal | Hrithik Rayapati | Garv Kalra

Dr Rahul Joshi
Signature of SIU Guide/Mentor

Date: 05/5/2024

Endorsement by

Place: SIT, Pune

Academic Integrity Committee (AIC)

Note: It is mandatory that the Similarity Index report of plagiarism (only first page) should be appended to the UG/PG dissertation

~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~