



**SECURE PAYMENT BY FACIAL RECOGNITION USING  
HAAR CASCADE ALGORITHM  
A PROJECT REPORT**

*Submitted By*

**MUGUNDHAN G (110821205029)**

**MUHILAN P (110821205030)**

**RAGHUL M (110820205038)**

*in partial fulfillment for the award of the degree*

*of*

**BACHELOR OF TECHNOLOGY**

*in*

**INFORMATION TECHNOLOGY**

**JAYA ENGINEERING COLLEGE, THIRUNINRAVUR**

**ANNA UNIVERSITY :: CHENNAI 600 025**

**MAY 2025**

**ANNA UNIVERSITY :: CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project report “**SECURE PAYMENT BY FACIAL RECOGNITION USING HAAR CASCADE ALGORTHIM**” is the bonafide work of, “**MUGUNDHAN G(110821205029), MUHILAN P(110821205030), RAGHUL M (110820205038)**” who carried out the project under my supervision.

**SIGNATURE**

**Dr.V.SEEDHA DEVI, Asso. Prof.,  
HEAD OF THE DEPARTMENT,  
Dept. of Information Technology,  
Jaya Engineering College,  
Thiruninravur,  
Chennai-602024**

**SIGNATURE**

**Dr.V.SEEDHA DEVI, Asso. Prof.,  
HEAD OF THE DEPARTMENT,  
Dept. of Information Technology,  
Jaya Engineering College,  
Thiruninravur,  
Chennai-602024**

Submitted for the viva voce examination held on \_\_\_\_\_

**INTERNAL EXAMINER**

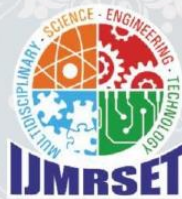
**EXTERNAL EXAMINER**

# CERTIFICATE OF PUBLICATION

## INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH IN SCIENCE, ENGINEERING AND TECHNOLOGY (IJMRSET)

*(A Monthly, Peer Reviewed, Referred, Multidisciplinary, Scholarly Indexed, Open Access & High Impact Factor Journal)*

**HIGH IMPACT FACTOR: 8.206**



**The Board of IJMRSET is hereby awarding this certificate to**

**MUGUNDHAN.G**

**UG Student, Dept. of IT, Jaya Engineering College, Chennai, Tamil Nadu, India**

***in Recognition of Publication of the Research Paper Entitled***

***“Secure Payment By Facial Recognition Using Haar  
Cascade Algorithm”***

***Published in IJMRSET, Volume 8, Issue 5, May 2025***



Crossref



INNO



SPACE

SJIF Scientific Journal Impact Factor

**ISSN** INTERNATIONAL  
STANDARD  
SERIAL  
NUMBER  
INDIA

**e-ISSN: 2582-7219**



*Jany*  
**Editor-in-Chief**

[www.ijmrset.com](http://www.ijmrset.com) ✉ [ijmrset@gmail.com](mailto:ijmrset@gmail.com)

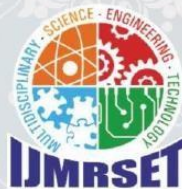


# CERTIFICATE OF PUBLICATION

## INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH IN SCIENCE, ENGINEERING AND TECHNOLOGY (IJMRSET)

*(A Monthly, Peer Reviewed, Referred, Multidisciplinary, Scholarly Indexed, Open Access & High Impact Factor Journal)*

**HIGH IMPACT FACTOR: 8.206**



**The Board of IJMRSET is hereby awarding this certificate to**

**MUHILAN.P**

**UG Student, Dept. of IT, Jaya Engineering College, Chennai, Tamil Nadu, India**

***in Recognition of Publication of the Research Paper Entitled***

**“Secure Payment By Facial Recognition Using Haar  
Cascade Algorithm”**

***Published in IJMRSET, Volume 8, Issue 5, May 2025***



Crossref



INNO



SPACE

SJIF Scientific Journal Impact Factor

**ISSN** INTERNATIONAL  
STANDARD  
SERIAL  
NUMBER  
INDIA

e-ISSN: 2582-7219



*[Signature]*  
**Editor-in-Chief**

[www.ijmrset.com](http://www.ijmrset.com) ✉ [ijmrset@gmail.com](mailto:ijmrset@gmail.com)

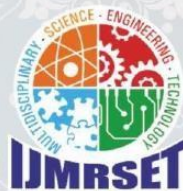


# CERTIFICATE OF PUBLICATION

## INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH IN SCIENCE, ENGINEERING AND TECHNOLOGY (IJMRSET)

(A Monthly, Peer Reviewed, Referred, Multidisciplinary, Scholarly Indexed, Open Access & High Impact Factor Journal)

**HIGH IMPACT FACTOR: 8.206**



**The Board of IJMRSET is hereby awarding this certificate to**

**RAGHUL.M**

**UG Student, Dept. of IT, Jaya Engineering College, Chennai, Tamil Nadu, India**

**in Recognition of Publication of the Research Paper Entitled**

**“Secure Payment By Facial Recognition Using Haar  
Cascade Algorithm”**

**Published in IJMRSET, Volume 8, Issue 5, May 2025**



Crossref



INNO



SPACE

SJIF Scientific Journal Impact Factor

**ISSN** INTERNATIONAL  
STANDARD  
SERIAL  
NUMBER  
INDIA

e-ISSN: 2582-7219



*Jany*  
Editor-in-Chief

www.ijmrset.com ✉ ijmrset@gmail.com



# JAYA ENGINEERING COLLEGE

APPROVED BY AICTE NEW DELHI, AFFILIATED TO ANNA UNIVERSITY - CHENNAI, ACCREDITED BY NAAC & NBA

THIRUNINRAVUR, CHENNAI - 602024



## DEPARTMENT OF IT, CSE & MCA International Conference on Innovative Technologies & Engineering



ICITE 2K25

### CERTIFICATE OF PARTICIPATION

This is to certify that Dr/Mr/Ms/Mrs. MUGUNDHAN . G / IT  
of JAYA ENGINEERING COLLEGE has  
participated in The International Conference on Innovative Technologies and  
Engineering - ICITE 2K25 and presented the Research Paper titled SECURE PAYMENT  
By FACIAL RECOGNITION USING HAAR CASCADE AND LOCAL BINARY PATTERN HISTOGRAM  
(LBPH).  
organized by Departments of IT, CSE and MCA held on 09/05/2025.

CONVENOR

Dr.V.SEEDHA DEVI HOD/IT

VICE PRINCIPAL

Dr. S.RAJENDRAN

PRINCIPAL

Dr.V.SURESHKUMAR





# JAYA ENGINEERING COLLEGE

APPROVED BY AICTE NEW DELHI, AFFILIATED TO ANNA UNIVERSITY - CHENNAI, ACCREDITED BY NAAC & NBA  
THIRUNINRAVUR, CHENNAI - 602024



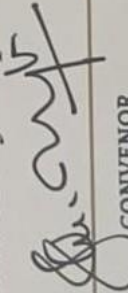
## DEPARTMENT OF IT, CSE & MCA International Conference on Innovative Technologies & Engineering


ICITE 2K25




## CERTIFICATE OF PARTICIPATION

This is to certify that ~~Dr~~/Mr/Ms/Mrs. MUHILAN.P/IT  
of JAYA ENGINEERING COLLEGE has  
participated in The International Conference on Innovative Technologies and  
Engineering - ICITE 2K25 and presented the Research Paper titled SECURE PAYMENT  
By FACIAL RECOGNITION USING HARR CASCADE AND LOCAL BINARY PATTERN HISTOGRAM  
(LBPH)  
organized by Departments of IT, CSE and MCA held on 09/05/2025.

  
CONVENOR  
Dr. V. SEEDHA DEVI HOD/IT

  
VICE PRINCIPAL  
Dr. S. RAJENDRAN

  
PRINCIPAL  
Dr. V. SURESHKUMAR



# JAYA ENGINEERING COLLEGE

APPROVED BY AICTE NEW DELHI, AFFILIATED TO ANNA UNIVERSITY - CHENNAI, ACCREDITED BY NAAC & NBA

THIRUNINRAVUR, CHENNAI - 602024



## DEPARTMENT OF IT, CSE & MCA International Conference on Innovative Technologies & Engineering



ICITE 2K25

## CERTIFICATE OF PARTICIPATION

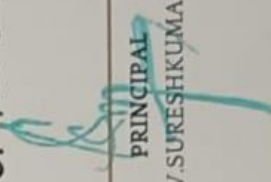
This is to certify that Dr/Mr/Ms/Mrs. RAGHUL M /IT  
of JAYA ENGINEERING COLLEGE has  
participated in The International Conference on Innovative Technologies and  
Engineering - ICITE 2K25 and presented the Research Paper titled SECURE PAYMENT  
By FACIAL RECOGNITION USING HAAR CASCADE AND LOCAL BINARY PATTERN  
organized by Departments of IT, CSE and MCA held on 09/05/2025. HISTOGRAM (LBPH).

  
CONVENOR

Dr. V. SEEDHA DEVI HOD/IT



VICE PRINCIPAL  
Dr. S. RAJENDRAN

  
PRINCIPAL  
Dr. V. SURESHKUMAR



## ABSTRACT

In today's fast-paced digital world, time is a crucial resource. Traditional payment methods—cash, credit cards, and mobile wallets—often involve several steps such as authentication, card insertion, PIN entry, and network processing, which can cause delays and inefficiencies. To address these issues, we propose a Face Recognition Payment System that enables users to make payments instantly through facial authentication, eliminating the need for physical cards or manual input. This system uses the Haar Cascade algorithm, a machine learning-based technique for rapid and accurate face detection. The process begins with user registration via a mobile application, where facial data and payment credentials are securely stored in a database. When a payment is initiated, the user's face is captured using a scanner and matched against the stored data. Upon successful verification, the payment is processed and a confirmation message is sent. The system enhances security by reducing risks associated with card fraud, PIN theft, and unauthorized transactions. It also provides a contactless, hands-free experience, which is especially useful in public spaces where hygiene and speed are essential. By integrating facial recognition into payment systems, transactions become faster, more secure, and more convenient. This approach has the potential to transform payments in retail, dining, transportation, and other commercial environments.

**Keywords** Artificial intelligence, Machine learning, Contactless payment, Haar Cascade, Biometrics, Secure payment, Fast and Convenient payment

## ACKNOWLEDEMENT

We profound gratitude to all individuals and institutions whose unwavering support and guidance have shaped my academic journey and contributed to the completion of this project

We Thank, **Anna University**, for providing an enriching academic environment, exceptional faculty, and valuable resources that have profoundly shaped my educational experience

Special thanks to the Management and The Principal of **Jaya Engineering College** for their strategic vision, continuous support, and commitment to institutional development

Our sincere appreciation to our beloved Head of the Department, Guided by **Dr.V.Seedha Devi, Associate Prof.,** Department of Information Technology for their visionary leadership, dedication to academic excellence, and unwavering support of students aspirations.

We extend our deepest thanks to all faculty members white expertise, encouragement and passion for teaching have inspired and empowered me throughout this project

Our heartfelt gratitude to our parents for their unwavering encouragement, sacrifices, and endless support throughout my educational journey.

We are graceful to our friends for their camaraderie, encouragement, and shared experiences, making this journey memorable and enjoyable.



## **TABLE OF CONTENTS**

<b>CHAPTER NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
	<b>ABSTRACT</b>	<b>iii</b>
	<b>LIST OF FIGURES</b>	<b>vi</b>
	<b>LIST OF NOTATIONS</b>	<b>viii</b>
	<b>LIST OF ABBREVIATIONS</b>	<b>ix</b>
<b>I</b>	<b>INTRODUCTION</b>	<b>1</b>
<b>II</b>	<b>FEASIBILITY STUDY</b>	<b>12</b>
	2.1 Existing system	12
	2.2 Proposed System	13
	2.3 System Requirement	15
	2.3.1 Hardware Description	15
	2.3.2 Software Description	16
<b>III</b>	<b>LITERATURE SURVEY</b>	<b>17</b>
<b>IV</b>	<b>SYSTEM DESIGN</b>	<b>26</b>
	4.1 System Architecture	26
	4.2 Use Case	27
	4.3 Modules	28
<b>V</b>	<b>SYSTEM IMPLEMENTATION</b>	<b>37</b>
	5.1 Sample Code	37
	5.2 Procedure for Execution	50
<b>VI</b>	<b>SYSTEM TESTING</b>	<b>51</b>
	6.1 Introduction	51
	6.2 Developing Methodologies	51
	6.3 Test Objectives	51
	6.4 Types of Testing	51
	6.5 Quality Assurance	54
<b>VII</b>	<b>CONCLUSION AND ENHACEMENT</b>	<b>59</b>
	<b>APPENDICES</b>	<b>60</b>

## LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO.
1.1	Components of Artificial Intelligence	1
1.2	Application of AI	3
1.2.1	Types of ML	7
1.2.2	Benefits of Machine Learning	7
1.3	Facial recognition	8
1.4	Computer Vision	10
2.1	Figure showing the features	14
2.2	Haar Cascades algorithm	14
4.1	System Architecture	26
4.2	Use Case Diagram	27
4.3	Class Diagram for User Dashboard Module	28
4.4	Sequence Diagram for User Dashboard Module	29
4.5	Collaboration Diagram for User Dashboard Module	30
4.6	Class Diagram for Scanner Payment Module	31
4.7	Sequence Diagram for Scanner Payment Module	32
4.8	Collaboration Diagram for Scanner Payment Module	33
4.9	Class Diagram for Shop Dashboard Module	34
4.10	Sequence Diagram for Shop Dashboard Module	35
4.11	Collaboration Diagram for Shop Dashboard Module	36
A1.1	Opening the face pay web app	60
A1.2	Running the face pay web app	60



A1.3	Interface of face pay web app	61
A1.4	Registering our data	61
A1.5	Face Capturing and Training	62
A1.6	Login Interface	62
A1.7	Login Page	63
A1.8	Opening the folder of scanner app	63
A1.9	Face capturing and wallet transferring	64
A1.10	Wallet transferred	64
A1.11	Shop website	65

## LIST OF NOTATIONS

NOTATION

NOTATION NAME



Actor



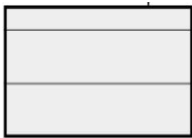
Use case



Message



Association



Class



Inheritance



## LIST OF ABBREVIATIONS

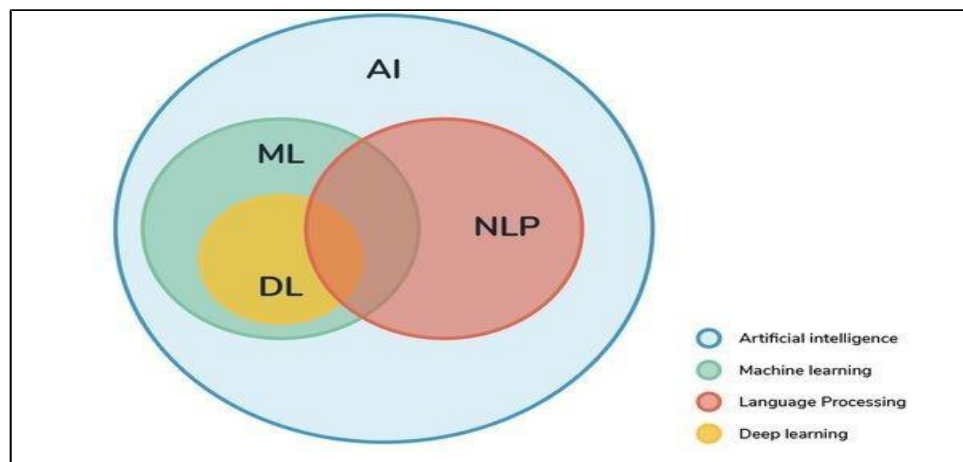
Abbreviation	Expansion
AI	Artificial Intelligence
API	Application Programming Interface
CCV	Card Code Verification
CNN	Convolutional Neural Network
CV	Computer Vision
DB	Database
DNN	Deep Neural Network
FR	Facial Recognition
GUI	Graphical User Interface
HTML	Hypertext Markup Language
ID	Identification
LBPH	Local Binary Patterns Histogram
ML	Machine Learning
OTP	One-Time Password
SQL	Structured Query Language

# CHAPTER I

## INTRODUCTION

### 1.1 Artificial Intelligence (AI)

The father of Artificial Intelligence, John McCarthy, it is “The science and engineering of making intelligent machines, especially intelligent computer programs”. Artificial intelligence is a field, which combines computer science and robust datasets, to enable problem-solving. It also encompasses sub-fields of machine learning and deep learning, which are frequently mentioned in conjunction with artificial intelligence. These disciplines are comprised of AI algorithms which seek to create expert systems which make predictions or classifications based on input data. Artificial Intelligence is a field, which combines computer science and robust datasets, to enable problem-solving. It also encompasses sub-fields of machine learning and deep learning, which are frequently mentioned in conjunction with artificial intelligence. These disciplines are comprised of AI algorithms which seek to create expert systems which make predictions or classifications based on input data. It is related to the similar task of using computers to understand human intelligence. One of the key components of AI is machine learning, which involves training algorithms on large datasets to recognize patterns and make predictions without being explicitly programmed. Deep learning, a subset of machine learning, uses artificial neural networks inspired by the structure and function of the human brain to achieve even greater levels of performance in tasks such as image and speech recognition, natural language processing, and autonomous decision-making.



*Fig 1.1 Components of Artificial Intelligence*

Machine learning, a critical component of AI, involves training algorithms on extensive datasets to recognize patterns and make predictions without explicit programming. This process enables machines to learn from data, adapt to new information, and improve performance over time. Deep learning, a subset of machine learning, utilizes sophisticated artificial neural networks inspired by the human brain's structure and functionality. These neural networks enable AI systems to achieve exceptional proficiency in tasks such as image and speech recognition, natural language processing, and autonomous decision-making. The pursuit of AI extends beyond traditional computing by seeking to replicate cognitive functions associated with human intelligence. This includes tasks like reasoning, problem-solving, learning, and understanding natural language. AI's impact spans across diverse industries, from healthcare and finance to transportation and entertainment, revolutionizing how businesses operate and how people interact with technology. Despite its transformative potential, AI presents ethical and societal challenges, including concerns about privacy, bias in algorithms, and the impact on employment. As AI technologies continue to evolve, interdisciplinary collaboration among researchers, policymakers, and industry leaders is essential to ensure responsible development and deployment of AI systems that benefit society while addressing potential risks.

### **1.1.1 Components of AI**

- **Machine Learning**

A subset of AI that enables machines to learn from data and improve their performance over time without being explicitly programmed.

- **Deep Learning**

A type of machine learning that utilizes artificial neural networks with multiple layers to process and learn from complex data representations.

- **Natural Language Processing (NLP)**

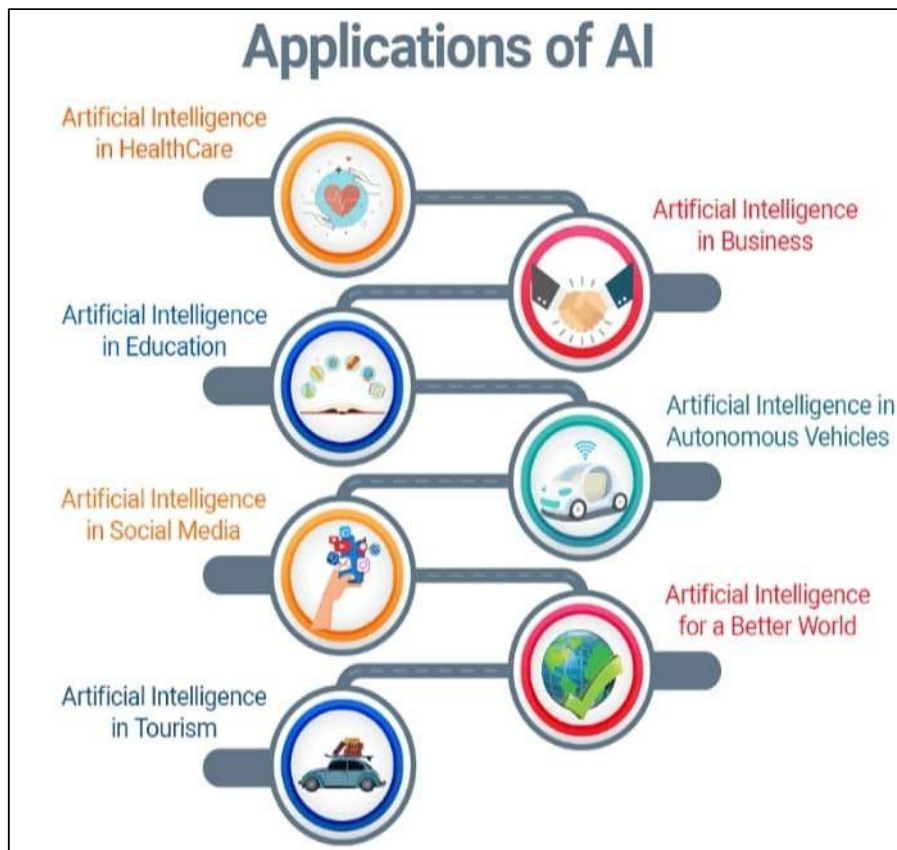
The branch of AI concerned with enabling computers to understand, interpret, and generate human language.

- **Computer Vision**

A field of AI focused on enabling computers to interpret and understand the visual world, including image recognition, object detection, and scene understanding.

- **Robotics**

The intersection of AI and robotics, aiming to create intelligent machines capable of performing physical tasks in diverse environments.



*Fig 1.2 Application of AI*

## **1.2 Machine Learning (ML)**

Machine Learning (ML) is a subset of artificial intelligence (AI) that represents a significant paradigm shift in how computers are programmed to perform tasks. Unlike traditional programming, where explicit instructions are provided to the computer to execute tasks, ML algorithms enable computers to learn and improve from experience without being explicitly programmed for each task. At the heart of ML lies the concept of learning from data. Instead of relying on predefined rules or instructions, ML algorithms learn patterns and relationships inherent in the data through statistical analysis and iterative optimization. This process involves training the algorithm on a dataset that contains examples of input features and corresponding output labels or targets. By analyzing this data, the algorithm learns to generalize from the training examples and



make predictions or decisions on new, unseen data. Machine learning (ML) is a discipline of artificial intelligence (AI) that provides machines with the ability to automatically learn from data and past experiences while identifying patterns to make predictions with minimal human intervention. Machine learning methods enable computers to operate autonomously without explicit programming. ML applications are fed with new data, and they can independently learn, grow, develop, and adapt. They derive insightful information from large volumes of data by leveraging algorithms to identify patterns and learn in an iterative process. ML algorithms use computation methods to learn directly from data instead of relying on any predetermined equation that may serve as a model. The performance of ML algorithms adaptively improves with an increase in the number of available samples during the ‘learning’ processes.

### 1.2.1 Types of Machine Learning

**Machine learning algorithms fall into five broad categories:** Supervised learning, unsupervised learning, semi-supervised learning, self-supervised and reinforcement learning.

#### 1. Supervised machine learning

Supervised machine learning is a type of machine learning where the model is trained on a labeled dataset (i.e., the target or outcome variable is known). For instance, if data scientists were building a model for tornado forecasting, the input variables might include date, location, temperature, wind flow patterns and more, and the output would be the actual tornado activity recorded for those days. Supervised learning is commonly used for risk assessment, image recognition, predictive analytics and fraud detection, and comprises several types of algorithms.

- **Regression algorithms:** They predict output values by identifying linear relationships between real or continuous values (e.g., temperature, salary). Regression algorithms include linear regression, random forest and gradient boosting, as well as other subtypes.
- **Classification algorithms:** They predict categorical output variables (e.g., “junk” or “not junk”) by labelling pieces of input data. Classification algorithms include logistic regression, k-nearest neighbours and support vector machines (SVMs), among others.
- **Naïve Bayes classifiers:** They enable classification tasks for large datasets. They’re also part of a family of generative learning algorithms that model the input distribution of a given

class or/category. Naïve Bayes algorithms include decision trees, which can actually accommodate both regression and classification algorithms.

- **Neural networks:** It simulate the way the human brain works, with a huge number of linked processing nodes that can facilitate processes like natural language translation, image recognition, speech recognition and image creation.
- **Random forest algorithms:** It predict a value or category by combining the results from a number of decision trees.

## 2. Unsupervised machine learning

Unsupervised learning algorithms like Apriori, Gaussian Mixture Models (GMMs) and principal component analysis (PCA)—draw inferences from unlabelled datasets, facilitating exploratory data analysis and enabling pattern recognition and predictive modelling. The most common unsupervised learning method is cluster analysis, which uses clustering algorithms to categorize data points according to value similarity (as in customer segmentation or anomaly detection). Association algorithms allow data scientists to identify associations between data objects inside large databases, facilitating data visualization and dimensionality reduction.

- **K-means clustering:** It assigns data points into K groups, where the data points closest to a given centroid are clustered under the same category and K represents clusters based on their size and level of granularity. K-means clustering is commonly used for market segmentation, document clustering, image segmentation and image compression.
- **Hierarchical clustering:** It describes a set of clustering techniques, including agglomerative clustering where data points are initially isolated into groups and then merged iteratively based on similarity until one cluster remains and divisive clustering where a single data cluster is divided based on the differences between data points.
- **Probabilistic clustering:** It helps solve density estimation or “soft” clustering problems by grouping data points based on the likelihood that they belong to a particular distribution.

## 3. Self-supervised machine learning

Self-supervised learning (SSL) enables models to train themselves on unlabelled data, instead of requiring massive annotated and/or labelled datasets. SSL algorithms, also called

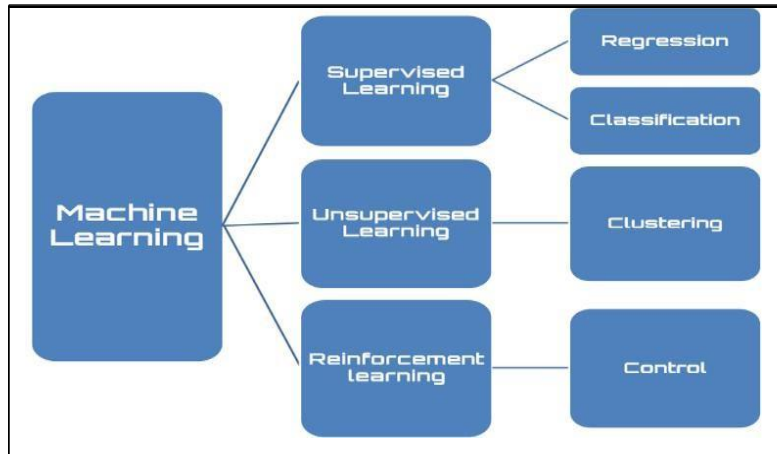
predictive or pretext learning algorithms, learn one part of the input from another part, automatically generating labels and transforming unsupervised problems into supervised ones. These algorithms are especially useful for jobs like computer vision and NLP, where the volume of labelled training data needed to train models can be exceptionally large (sometimes prohibitively so).

#### **4. Reinforcement learning**

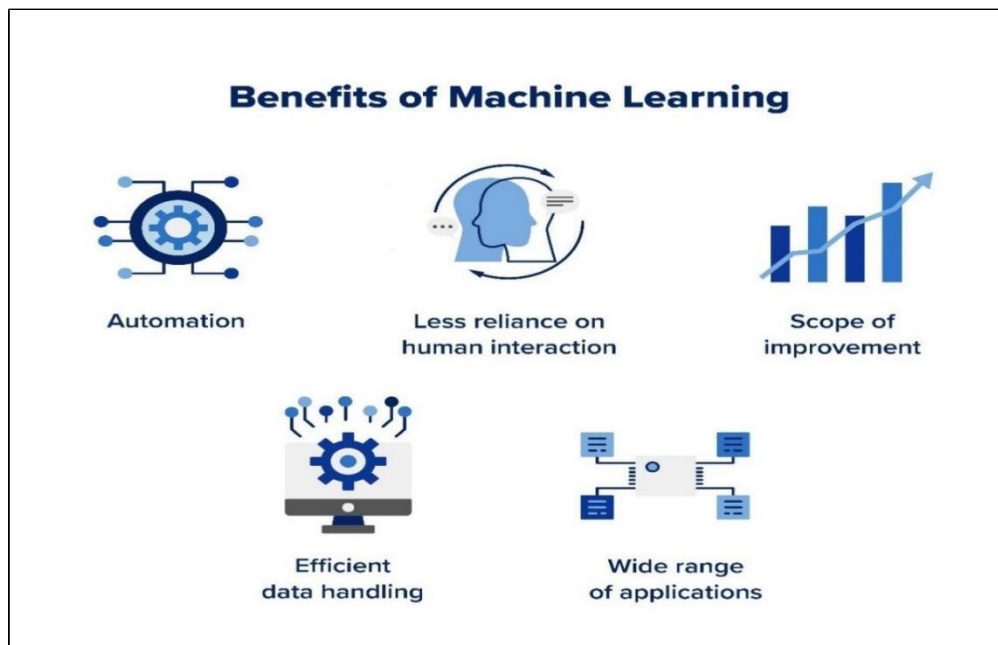
Reinforcement learning, also called reinforcement learning from human feedback (RLHF), is a type of dynamic programming that trains algorithms using a system of reward and punishment. To deploy reinforcement learning, an agent takes actions in a specific environment to reach a predetermined goal. The agent is rewarded or penalized for its actions based on an established metric (typically points), encouraging the agent to continue good practices and discard bad ones. With repetition, the agent learns the best strategies. Reinforcement learning algorithms are common in video game development and are frequently used to teach robots how to replicate human tasks.

#### **5. Semi-supervised learning**

The fifth type of machine learning technique offers a combination between supervised and unsupervised learning. Semi-supervised learning algorithms are trained on a small labelled dataset and a large unlabelled dataset, with the labelled data guiding the learning process for the larger body of unlabelled data. A semi-supervised learning model might use unsupervised learning to identify data clusters and then use supervised learning to label the clusters. Generative adversarial networks (GANs) deep learning tool that generates unlabelled data by training two neural networks are an example of semi-supervised machine learning. Regardless of type, ML models can glean data insights from enterprise data, but their vulnerability to human/data bias make responsible AI practices an organizational imperative.



*Fig 1.2.1 Types of ML*



*Fig 1.2.2 Benefits of Machine Learning*

### 1.3 Facial Recognition

Facial recognition is a biometric technology that identifies or verifies a person by analysing their facial features. It works by capturing an image or video of a face, mapping key facial landmarks (such as the distance between the eyes, nose shape, jawline, etc.), and comparing it to a stored database of faces.

#### **How Facial Recognition Works:**

1. **Face Detection:** The system detects a face in an image or video.



2. **Feature Extraction:** The system maps facial features and creates a unique representation (faceprint).
3. **Comparison:** The faceprint is compared with stored templates in the database.
4. **Verification/Identification** If a match is found, the system either verifies the identity (one-to-one match) or identifies the person (one-to-many match).

**Uses of Facial Recognition:**

- Security & Surveillance (law enforcement, airport security)
- Authentication & Access Control (smartphones, banking apps, door unlocking)
- Payments & Transactions (face recognition payment systems)
- Personalized User Experience (social media filters, targeted advertising)
- Healthcare & Attendance Tracking (hospital patient identification, employee check-in)



*Fig 1.3 Facial recognition*

## **1.4 Computer Vision**

Computer vision is a field of computer science that focuses on enabling computers to identify and understand objects and people in images and videos. Like other types of AI, computer vision seeks to perform and automate tasks that replicate human capabilities. In this case, computer vision seeks to replicate both the way humans see, and the way humans make sense of what they see. The range of practical applications for computer vision technology makes it a central component of many modern innovations and solutions. Computer vision can be run in the cloud or on premises.

### **1.4.1 How computer vision works**

Computer vision applications use input from sensing devices, artificial intelligence, machine learning, and deep learning to replicate the way the human vision system works. Computer vision

applications run on algorithms that are trained on massive amounts of visual data or images in the cloud. They recognize patterns in this visual data and use those patterns to determine the content of other images.

### 1.4.2 Key Steps in Computer Vision

1. **Image Acquisition:** The system captures an image or video using a camera or sensors.
2. **Preprocessing:** The image is enhanced (grayscale conversion, noise reduction, resizing, etc.) to improve accuracy.
3. **Feature Extraction:** The system identifies key features (edges, colours, shapes, textures, etc.) to understand the content.
4. **Object Recognition & Classification:** Machine learning or deep learning models analyse the extracted features to classify objects (e.g., identifying a face in face recognition).
5. **Decision Making & Output:** The system makes predictions based on the analysis and triggers an action (e.g., unlocking a phone, detecting defects in manufacturing, self-driving car navigation).

### 1.4.3 Computer Vision Techniques:

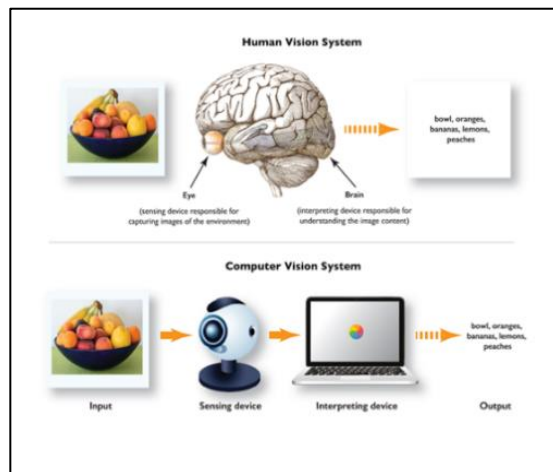
- **Image Classification:** Identifies the category of an image (e.g., cat vs. dog).
- **Object Detection:** Locates objects in an image (e.g., detecting people in a crowd).
- **Face Recognition:** Identifies people based on facial features (like in your payment system).
- **Pose Estimation:** Detects human body positions and movements.
- **Optical Character Recognition (OCR):** Extracts text from images (e.g., scanning documents).
- **Image Segmentation:** Divides an image into meaningful parts (used in medical imaging).

### 1.4.4 How This Relates to Your Face Recognition Payment System:

Since you're working on a **face recognition payment system**, computer vision will play a crucial role in:

- **Detecting faces** in a live camera feed (Haar cascade in your case).
- **Extracting facial features** for recognition.

- **Matching** the detected face with stored data for authentication.
- **Triggering payment processing** upon successful recognition.



*Fig 1.4 Computer Vision*

## 1.5 Haar Cascade Algorithm

The **Haar Cascade Algorithm** is an object detection method used to identify faces and other objects in images and videos. It is based on **Haar-like features** and was introduced by **Viola and Jones (2001)** as an efficient face detection technique.

### 1.5.1 How Haar Cascade Works:

#### 1. Haar-like Features Extraction

- The algorithm analyses rectangular regions in an image to detect patterns like edges and textures.
- It uses features similar to Haar wavelets, such as **edge detection, line detection, and four-rectangle features**.

#### 2. Integral Image Calculation

- To speed up feature computation, Haar cascades use an **integral image** (a summed-area table) that allows quick calculations of pixel sums in rectangular regions.

### 3. Ada-boost Training (Feature Selection)

- Thousands of Haar features are tested, but only the most relevant ones are selected using the **Ada-boost** machine learning algorithm.
- Ada-boost assigns **higher weights** to features that perform well in distinguishing faces from non-faces.

### 4. Cascade Classifier (Multi-stage detection)

- Instead of processing the entire image at once, Haar cascades use a **cascade of weak classifiers** to eliminate non-faces early.
- The image is scanned at multiple scales (pyramid approach) to detect faces of different sizes.

#### 1.5.2 Advantages of Haar Cascade

- **Fast and Efficient:** Uses an integral image for rapid calculations.
- **Lightweight:** Runs well on low-power devices.
- **Real-time Detection:** Can process video frames efficiently.

#### 1.5.3 Limitations

- **Prone to False Positives:** May detect non-face objects as faces.
- **Sensitive to Lighting & Angle:** Works best with frontal face images.
- **Not as Accurate as Deep Learning:** Struggles with complex variations (e.g., occlusions, extreme angles).



## CHAPTER II

### FEASIBILITY STUDY

#### 2.1 Existing System

The popularity of contactless payment systems has significantly increased, especially after the COVID-19 pandemic. These systems offer numerous benefits for both public health and user convenience. Among the various contactless payment methods, facial recognition payment is one of the latest innovations. As the name suggests, this technology enables users to make transactions simply by scanning their faces, eliminating the need for cash, credit/debit cards, or even a mobile phone. With its speed, ease of use, and enhanced security. One of its key advantages is security—while traditional payment methods can be compromised through stolen cards or hacked credentials, a person's face is unique and cannot be easily replicated. However, to ensure reliability, the system must be sophisticated enough to distinguish between a real person and a photograph or video, preventing fraud attempts.

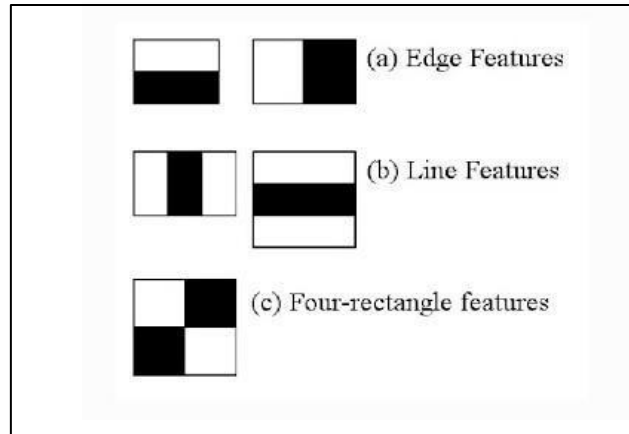
##### 2.1.1 Limitations

- **Privacy concerns:** Facial recognition technology captures sensitive biometric data, raising concerns about how this information is stored, accessed, and protected from unauthorized use.
- **Accuracy issues:** Facial recognition systems can misidentify individuals due to factors like poor lighting, camera angle, facial expressions, or variations in appearance over time.
- **Bias in identification:** Studies have shown that facial recognition algorithms can be biased against certain ethnicities or genders, leading to unfair or inaccurate results.
- **Potential for abuse:** Malicious actors could potentially use facial recognition technology to track individuals, impersonate others, or commit fraud.
- **Lack of user control:** Individuals may not have full control over how their facial data is collected, stored, and used.
- **Security vulnerabilities:** Facial recognition systems can be susceptible to hacking or manipulation, allowing unauthorized access to payment information.

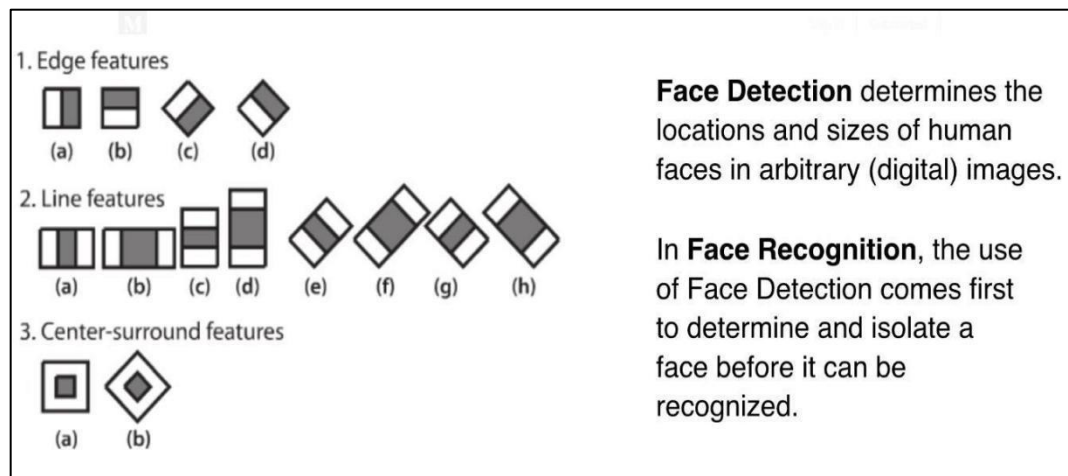
- **Accessibility issues:** Individuals with certain facial features or disabilities may struggle to be accurately recognized by the system.
- **Ethical concerns:** The use of facial recognition technology raises ethical questions about the balance between convenience and individual privacy.

## 2.2 Proposed System

The system ensures fast, secure transactions without cash, cards, or phones, using AI-driven liveness detection to prevent fraud. Biometric data is encrypted for privacy, with MFA adding extra security. Users can manage, update, or delete their data anytime. Cross-platform support ensures compatibility across devices and networks. Adaptive AI enhances accuracy, while compliance with GDPR, CCPA, and other laws ensures ethical standards. Secure authentication protocols and spoof detection prevent identity theft. Continuous monitoring detects anomalies, and bias is mitigated through diverse datasets. Accuracy improves with high-quality cameras and real-time processing. Regular updates address security vulnerabilities, and transparency policies ensure ethical AI use. Additional security includes homomorphic encryption, federated learning for bias reduction, and adaptive recognition for evolving user appearances. Blockchain-based identity verification reduces breaches, and multi-layered authentication secures high-risk transactions. Hardware-level security and ethical AI oversight further enhance protection, with public awareness campaigns promoting transparency. This uses machine learning techniques to get a high degree of accuracy from what is called “training data”. Haar Cascades use the Ada-boost learning algorithm which selects a small number of important features from a large set to give an efficient result of classifiers. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, haar features shown in below image are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under white rectangle from sum of pixels under black rectangle.



*Fig 2.1 Figure showing the features*



*Fig 2.2 Haar Cascades algorithm*

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

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

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

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

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

Cascade classifier training requires a set of positive samples and a set of negative images. You must provide a set of positive images with regions of interest specified to be used as positive samples. You can use the Image to label objects of interest with bounding boxes. The Image Labeler outputs a table to use for positive samples. You also must provide a set of negative images from 11 which the function generates negative samples automatically. To achieve acceptable detector accuracy, set the number of stages, feature type, and other function parameters.

## 2.3 System Requirements

### 2.3.1 Hardware Requirements

#### 1. For Payment Terminal (Scanner)

- **Camera:** HD/IR camera (min **720p resolution**)
- **Processing Unit:** Raspberry Pi / PC with **GPU support**

#### 2. For Backend Server

- **CPU:** Intel Core i5/i7 or equivalent (or ARM-based for Raspberry Pi)
- **RAM:** Minimum 8GB (16GB for better performance)

#### 3. For App

- Camera with minimum 5mp



## 2.3.2 Software Requirements

- **Server (Cloud-based or On-Premises - Linux/Windows):** The server hosts the backend application and handles all processing. It can be deployed on a cloud platform (like AWS, Google Cloud) or a local on-premises machine running Linux or Windows, depending on availability and scalability needs.
- **Programming Language (Python):** Python is the core programming language used in the system. It powers the backend logic, face detection algorithms, API creation, and database interactions due to its simplicity and strong library support. Face Detection: OpenCV (Haar Cascade algorithm)
- **Backend Framework (Flask):** Flask is a lightweight Python web framework used to build APIs. It connects the mobile app to backend services and handles requests for face detection, data submission, and responses.
- **Database (MySQL):** MySQL is used to store structured data such as user details, face recognition metadata, and logs. It enables efficient data management and querying.
- **Libraries:** OpenCV, face-recognition, NumPy, etc.
- **OpenCV:** OpenCV (Open-Source Computer Vision Library) is an open-source software library designed for real-time computer vision and image processing. It provides a wide range of tools and functions to perform tasks such as object detection, face recognition, image filtering, motion tracking, and camera calibration. Written in C++, it also offers bindings for Python, Java, and other languages, making it accessible to developers across different platforms. OpenCV is widely used in fields like robotics, artificial intelligence, and augmented reality, helping machines to interpret and understand visual data effectively.
- **Face-recognition:** It is a biometric technology that identifies or verifies a person's identity by analysing their facial features. It involves detecting a face in an image or video, extracting key facial landmarks (like the eyes, nose, and mouth), and comparing them with stored facial data to find a match. The process usually consists of three main steps: face detection, feature extraction, and face matching. Commonly used in security systems, smartphone unlocking, and payment systems, face recognition is powered by computer vision and machine learning techniques, often implemented using tools like OpenCV and deep learning models
- **NumPy:** Supports numerical operations needed during image processing

## CHAPTER III

### LITERATURE SURVEY

**3.1 Yongping Zhong, Hee-Cheol Moon "Investigating Customer Behavior of Using Contactless Payment in China: A Comparative Study of Facial Recognition Payment and Mobile QR-Code Payment", Sustainability - MDPI, ISSN: 2071-1050, Volume 14, issued date: June 10, 2022**

#### **Abstract**

This study investigates the impact of contactless payment methods, particularly facial recognition payment and mobile QR-code payment, on consumer behavior in China. The research highlights how perceived usefulness, ease of use, and security influence users' satisfaction and adoption behaviors. It finds that while both payment methods enhance digital transactions, QR-code payments are preferred due to habit formation and reliability, whereas facial recognition payment faces challenges related to security concerns and system accuracy. The study provides insights into improving contactless payment adoption and enhancing digital transaction experiences

#### **Description**

The research examines the behavioral differences between users of facial recognition and QR-code payments, analysing factors that drive continued usage. It finds that QR-code payments have a stronger influence on user satisfaction and habitual behavior. Many consumers initially try facial recognition payments but revert to QR codes due to privacy concerns and technical failures, such as issues with mask detection. The study suggests strategies for improving facial recognition payment adoption, including enhanced security measures and better consumer education

#### **Limitations**

- **Sample Diversity:** The study primarily focuses on younger users, limiting its generalizability to older demographics.
- **Technological Barriers:** Issues such as difficulty recognizing masked users and system errors affect the adoption of facial recognition payment.
- **Habitual Behavior:** Many users are accustomed to QR-code payments, making it challenging to shift behaviors.
- **Security Concerns:** Users fear data breaches and biometric information misuse.

- **Limited Comparative Analysis:** The study does not deeply compare facial recognition payments with other digital payment methods

**3.2** Marta Beltrán, Miguel Calvo, “**A privacy threat model for identity verification based on facial recognition**”, ELSEVIER - Computers & Security, ISSN: 0167 - 4048, Volume: 132, Issue: June 6, 2023

## **Abstract**

The proliferation of different types of photographic and video cameras makes it relatively simple and non-intrusive to acquire facial fingerprints with sufficient quality to perform individuals' identity verification. In most democratic societies, a debate has been occurring regarding using such techniques in different application domains. Discussions usually revolve around the trade - offs between utility (security in access control, mobile phone unlocking, payment processing, etc.), usability or economic gain and risks to citizens' rights and freedoms (privacy) or ethics. This paper identifies the common aspects of different solutions for identity verification based on facial recognition techniques within different application domains. It then performs a privacy threat modelling based on these common aspects to identify the most critical risk factors and a minimum set of safeguards to be considered for their management.

## **Description**

The paper, titled "A Privacy Threat Model for Identity Verification Based on Facial Recognition," explores the growing adoption of facial recognition technology for identity verification across various domains, such as access control, mobile authentication, and payment systems. While these technologies offer enhanced security and convenience, they raise significant privacy concerns regarding user data protection, unauthorized surveillance, and ethical implications. This research proposes a privacy threat model that systematically identifies critical risk factors associated with facial recognition-based identity verification. The model categorizes threats based on linkability, identifiability, non-repudiation, detectability, disclosure of information, unawareness, and non-compliance. The study further presents a taxonomy of application domains using facial recognition and outlines safeguards to mitigate privacy risks. The findings aim to enhance awareness of privacy vulnerabilities and encourage the implementation of robust security measures to ensure ethical and responsible use of facial recognition technologies.

## Limitations

- **Data Sensitivity:** The study acknowledges the challenges of securing biometric data, as facial recognition involves personally identifiable information that, if compromised, could have long-term consequences.
- **Regulatory Gaps:** Existing legal frameworks may not fully address the ethical and privacy concerns of facial recognition, leading to potential misuse.
- **Bias and Accuracy Issues:** Facial recognition systems may exhibit racial, gender, and age-related biases, affecting the accuracy and fairness of identity verification.
- **Security Vulnerabilities:** The study highlights the risk of spoofing attacks, presentation attacks (e.g., using masks or deepfakes), and database breaches.
- **Limited User Awareness:** Many users may not fully understand the privacy implications of facial recognition, leading to uninformed consent and reduced trust in these systems.
- **Dependence on Centralized Databases:** The use of centralized facial recognition databases increases the risk of large-scale data breaches and unauthorized access.
- **Collaboration Risks:** The involvement of multiple third-party entities in facial recognition-based identity verification complicates data security and compliance with privacy regulations

**3.3** Romano Araujo, Adarsh Potekar, Mayuresh Raikar, Sumedha Mainkar, Salil Salgaonkar, Ashish Narvekar, Yogini Lamgaonkar “**Face Recognition Based Payment Processing System**”, International Research Journal of Engineering and Technology (IRJET), e-ISSN: 2395-0056, p-ISSN: 2395-0072, Volume: 09, Issued: July, 2022

## Abstract:

The use of many cards' transaction in various places like shopping, restaurants, lodges and online transaction in regarding booking hotels, movies, flights and train tickets etc., are on a rise each second. Hence the problem of people carrying payment card & cash along with themselves and keep the cards/cash secure to use it all the time; chances of being victim of theft. As in present work, biometric facial recognition transactions are used everywhere. Use of One-Time Password (OTP) to secure the transaction further helps us in avoiding need of memorizing different passwords. Face recognition transaction systems are safe and secure with ease. It's known for its reliability and more efficient compared to other payment technics. A general idea of an online

transaction system using facial recognition is suggested. The technique chosen for facial identification are Local Binary Pattern Histogram (LBPH) and HAAR CASCADE.

### **Description**

The paper "Face Recognition Based Payment Processing System" explores the development of a biometric-based transaction system that utilizes facial recognition for secure and seamless payment processing. Traditional payment methods, such as credit/debit cards and passwords, pose security risks like theft and fraud. To address these issues, the proposed system integrates Haar Cascade and Local Binary Pattern Histogram (LBPH) algorithms for accurate facial recognition. The system captures a user's facial image, processes it to extract key features, and matches it with a pre-registered database to verify identity. Upon successful recognition, a One-Time Password (OTP) is sent to the user's registered mobile number to further enhance transaction security. The project is implemented using OpenCV and Python, ensuring efficient real-time processing. The study highlights the benefits of facial recognition in payment systems, including enhanced security, ease of use, and reduced dependency on physical payment methods. This research contributes to the advancement of biometric authentication in financial transactions by offering a secure, user-friendly alternative to conventional payment mechanisms.

### **Limitations**

- **Accuracy Challenges:** The system's effectiveness depends on lighting conditions, facial expressions, and obstructions (e.g., masks, glasses), which may affect recognition accuracy.
- **Security Concerns:** Despite encryption and OTP verification, facial spoofing attacks using photos or videos remain a potential risk.
- **Processing Speed:** Real-time face detection and authentication require high computational power, which may lead to delays in processing transactions on low-end devices.
- **Database Dependency:** The system relies on a pre-registered database, meaning that new users must enroll beforehand, limiting instant usability for first-time customers.
- **Privacy Issues:** Users may have concerns about biometric data storage and misuse, necessitating strong encryption and regulatory compliance.
- **Limited Scalability:** The system may face challenges in handling large-scale deployments, especially in crowded environments where multiple users attempt authentication simultaneously.

- **Internet Requirement:** OTP-based verification requires internet or mobile network connectivity, which can be a barrier in areas with poor network access

**3.4** Mohammad Fauzi Aziz, Gavin Sadiya Taraka, Sidharta Sidharta, “**Face Recognition as Base Protocol in Online Transactions**”, ELSEVIER - ScienceDirect, Volume: 245, ISSN: 1877-0509, Issue: 2024

### **Abstract**

From each to each year, from decades and decades, the fraud and scam are growing more and more powerful, although we cannot avoid it 100% since the fraud and scams using any method to make a crack of hole or other chances to steal that make a company and other e-commerce suffer losses in small or large numbers. This leads to new rules that can make e-commerce stand a chance and avoid the losses of either the e-commerce itself or the client that is trying to make online transactions. This paper has a goal to answer 5 research questions about face recognition as base protocol in online transaction. Method that we use is systematic literature review that has 7 steps from deciding research question until data synthesis. For the result from 5 research questions are answered very well, which is RQ1. How effective face recognition can decrease fraud identity on online transactions. RQ2. What kind of struggle and problem makes this technology unable to integrate with online payment. RQ3. What kind of perception and the acceptance of the user about the face recognition method for authentication in online transactions? RQ4. How far the technology is for face recognition can integrate with the system online payment transaction that increases the efficiency transaction with or without compromising security. RQ5. What kind of law of implication and etiquette from the user face recognition in online transactions and how can the system be made to follow the regulation that already exists. From that our conclusion is clear which is face recognition is safe and secure but since the technology is very well and good, then need proper technology to support it further.

### **Description**

The paper "Face Recognition as Base Protocol in Online Transactions" explores the role of facial recognition technology in enhancing the security of digital transactions. With the increasing threats of fraud and identity theft in e-commerce, traditional authentication methods such as passwords and PINs are becoming more vulnerable. This study investigates how facial recognition can serve as a primary authentication protocol to reduce fraud risks and improve the efficiency of online payments.



**Using a systematic literature review approach, the research addresses five key questions:**

1. Effectiveness of facial recognition in reducing fraud in online transactions.
2. Challenges preventing seamless integration of facial recognition with digital payments.
3. User perceptions and acceptance of facial authentication.
4. Integration of face recognition technology with payment systems for improved security and efficiency.
5. Legal and ethical considerations surrounding biometric authentication.

The study concludes that while facial recognition is a secure and promising technology, it requires proper technological support, legal frameworks, and user education to achieve widespread adoption. The findings emphasize the need for multi-factor authentication, regulatory compliance, and continued advancements in AI-based face detection algorithms to enhance security and user trust

### **Limitations**

- **Security Risks:** Although facial recognition enhances security, it is susceptible to spoofing attacks (e.g., deepfakes, mask-based fraud).
- **Privacy Concerns:** Users may be unaware of how their biometric data is stored and used, leading to trust issues.
- **Legal & Ethical Issues:** The absence of global regulations on biometric data usage creates challenges in compliance and implementation.
- **Accuracy Variations:** Factors such as lighting, facial expressions, aging, and occlusions (e.g., masks, glasses) may reduce recognition accuracy.
- **Integration Challenges:** Many online payment platforms lack the infrastructure to support facial recognition as a standalone authentication method.
- **User Acceptance:** Some users may resist adopting facial recognition due to concerns over privacy, reliability, and usability.
- **Computational Requirements:** Real-time facial recognition demands high processing power and storage, which may not be feasible for all devices and networks.

**3.5** Kuldeep Vayadande, Rachit Chandawar, Anuj Mahajan, Swapnil Garud, Mansi Parse, Jaykumar Gavitt, Pushkar Gajdhane, Aryan Chalpe, Pratik Davare, Atharva Borade, Eshan Dasarwar, **“Optimized Haar-cascade Algorithm for Face Recognition and Authentication”**, International Journal of Intelligent and Application in Engineering, Volume 12, ISSN: 2147-6799, issue: 2024

### **Abstract**

What amuses biometrics is that it can be incorporated into various situations. Particularly, it can be used to conduct identification, security, and Internet security. A newly emerged biometric technology is related to 3D imagery of facial features while delivering additional flow of volume to empirical verification of human faces as well as identification. Conversely, it is very difficult to manipulate with large amount of 3D faces data that may come with some problems including the dimensionality reduction issues problem fighting with. This process serves to define face recognition as a viable means of personal identification nowadays, typically used during security, human–computer interaction systems, among others, and thus opting for more efficient algorithms for embedded systems is no longer negotiable. The smallest platforms are now utilizing the concept of embedded face recognition as the latest trendsetter. Lastly, neural networks are being potentially integrated into the system as they are considered to give the artificial systems both the speed and the accuracy. The canvas combines the concept of which includes LBP and MTCNN that brings all the edges to an end. Despite the fact that the PCA method before had brought about success in the dimensionality reduction sphere, a manifold theory has now become the trend in the process of handling facial expressions which have added complexity to the situation. The document explains the applications of the Haar cascade and face detection components, which includes before processing grayscale images up to after processing like integration with the facial recognition system. Furthermore, the course is video-academic and practical in that during the duration it deals with real time applications and issues and ethics principles which are mostly internalized to a student. The algorithms Library which is built on the abilities of face detection is the main tool of the developers and the researchers, on the one hand, that are used not only for face detection and employment of it but also for diversification of the technology, on the other hand. Progress is after all implemented in a double sense. thus, while the definition exposes the issues of ethics and imprecision, the main problem with computer vision cannot be forgotten either. Further, this result-based rescue approach also delivers the idea that the university is an getting prepared university as

it aids its students to expand relevant skills and knowledge that actually follow on every campus, contemporary society.

### **Description**

The paper "Optimized Haar-Cascade Algorithm for Face Recognition and Authentication" explores an improved approach to face detection and authentication using the Haar-Cascade algorithm. The study emphasizes the role of biometric security in modern applications such as identity verification, digital transactions, and surveillance systems. Traditional face recognition techniques often face challenges related to accuracy, speed, and adaptability to varying environmental conditions. To enhance performance, the proposed system integrates Local Binary Pattern (LBP) and Multi-Task Cascaded Convolutional Networks (MTCNN) with the Haar-Cascade classifier. This hybrid approach improves feature extraction, detection accuracy, and real-time processing. The system is tested in real-world scenarios, such as payment authentication, security access control, and human-computer interaction systems. The paper further discusses the application of machine learning and neural networks in face recognition, highlighting their role in improving facial feature identification and classification. The study concludes that the optimized algorithm achieves higher accuracy and faster processing than conventional Haar-Cascade models, making it suitable for embedded systems and mobile applications.

### **Limitations**

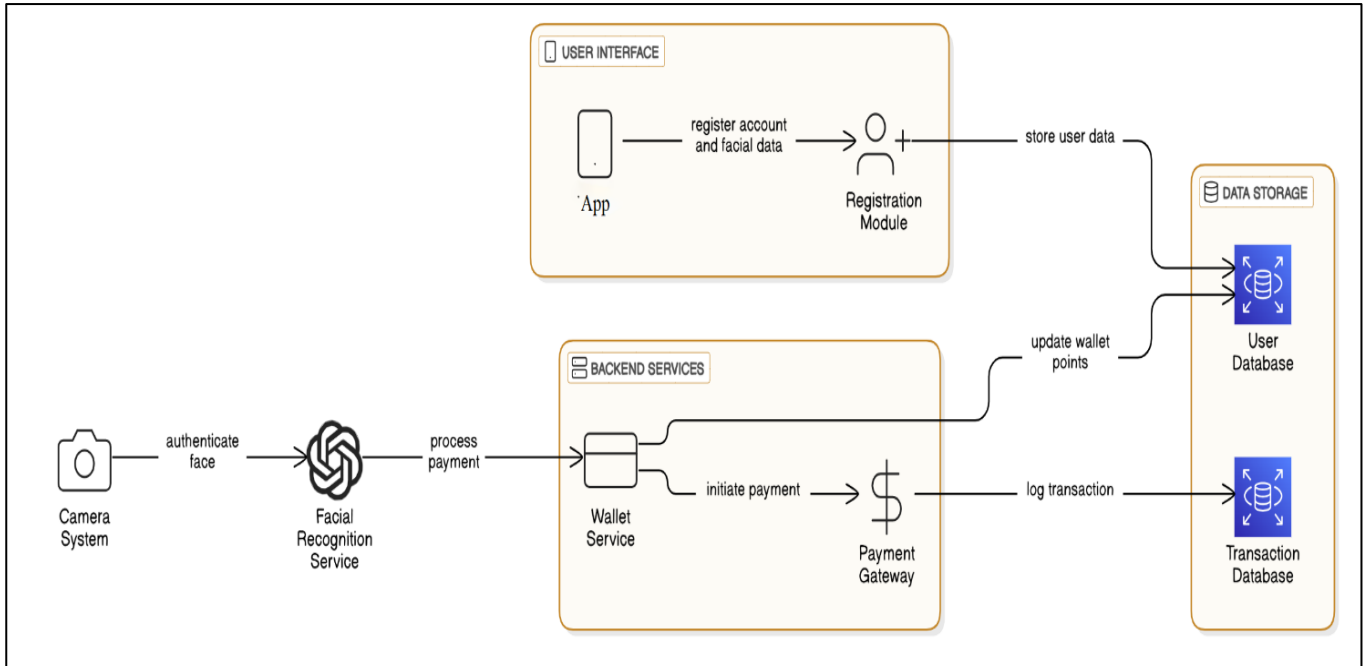
- **Sensitivity to Environmental Factors:** The system's performance is affected by lighting conditions, facial occlusions (e.g., masks, glasses), and background noise.
- **Security Risks:** Despite improvements, the algorithm remains vulnerable to spoofing attacks using high-resolution images or deepfake technology.
- **Computational Load:** While optimized, Haar-Cascade still requires significant computational power for large-scale applications, making real-time processing challenging on low-end devices.
- **Dependence on Predefined Features:** The model relies on pre-trained facial features, limiting adaptability to diverse demographics and unique facial structures.
- **Limited Accuracy Compared to Deep Learning Models:** While effective, the Haar-Cascade algorithm is less robust than deep learning-based recognition systems like CNNs or Transformers.

- **Scalability Challenges:** The system may struggle to maintain accuracy and efficiency when deployed in high-traffic environments, such as airports or crowded public places.
- **Data Privacy Concerns:** Storing biometric data for authentication raises ethical and legal concerns, requiring secure encryption and compliance with data protection regulations.

## CHAPTER IV

### SYSTEM DESIGN

#### 4.1 System Architecture

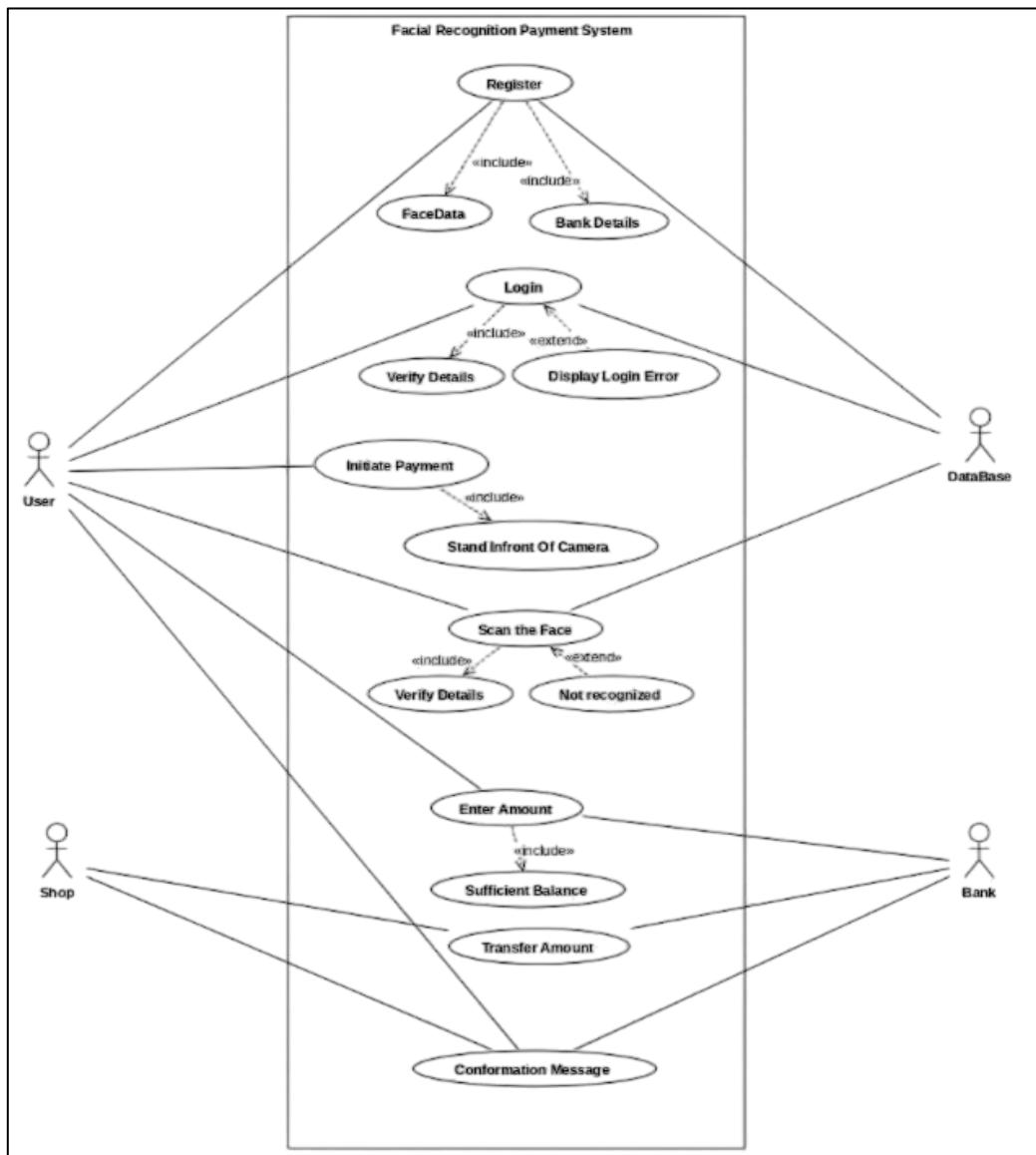


*Fig 4.1 System Architecture*

#### Description

The image presents a structured architecture diagram of a Face Recognition Payment System, illustrating the flow of data between different modules. It consists of four primary components: User Interface, Facial Recognition, Backend Services, and Data Storage. The User Interface includes a application where users register their accounts and facial data through a Registration Module, which stores the information in a User Database. The Facial Recognition Module involves a Camera System that captures the user's face for authentication, which is then verified by a Facial Recognition Service before processing the payment. Once authenticated, the Backend Services initiate a transaction through the Wallet Service, deducting the amount and updating the user's wallet balance. The transaction details are sent to a Payment Gateway, which processes the payment and logs the details in a Transaction Database for record-keeping. The Data Storage Module ensures that all user data, wallet balances, and payment transactions are securely maintained. The diagram effectively demonstrates the interaction between these modules, highlighting the secure and seamless process of facial recognition-based transactions.

## 4.2 Use Case Diagram



*Fig 4.2 Use Case Diagram*

### Description

The Use Case Diagram represents a Facial Recognition Payment System, illustrating interactions between key actors: the User, Database, Shop, and Bank. The User can register by providing Face Data, Bank Details, and optionally Fingerprint Data. During Login, the system verifies details and may display a Login Error if authentication fails. To initiate a payment, the user stands in front of a camera for Face Scanning, which includes verifying details and may extend to a Not Recognized scenario if authentication fails. After successful recognition, the user enters the payment amount, and the system checks for Sufficient Balance before transferring the amount.



Finally, a Confirmation Message is sent to complete the transaction. The Database stores user details, while the Bank verifies and processes payments. The Shop interacts with the system for payment processing, ensuring a secure and seamless biometric payment experience.

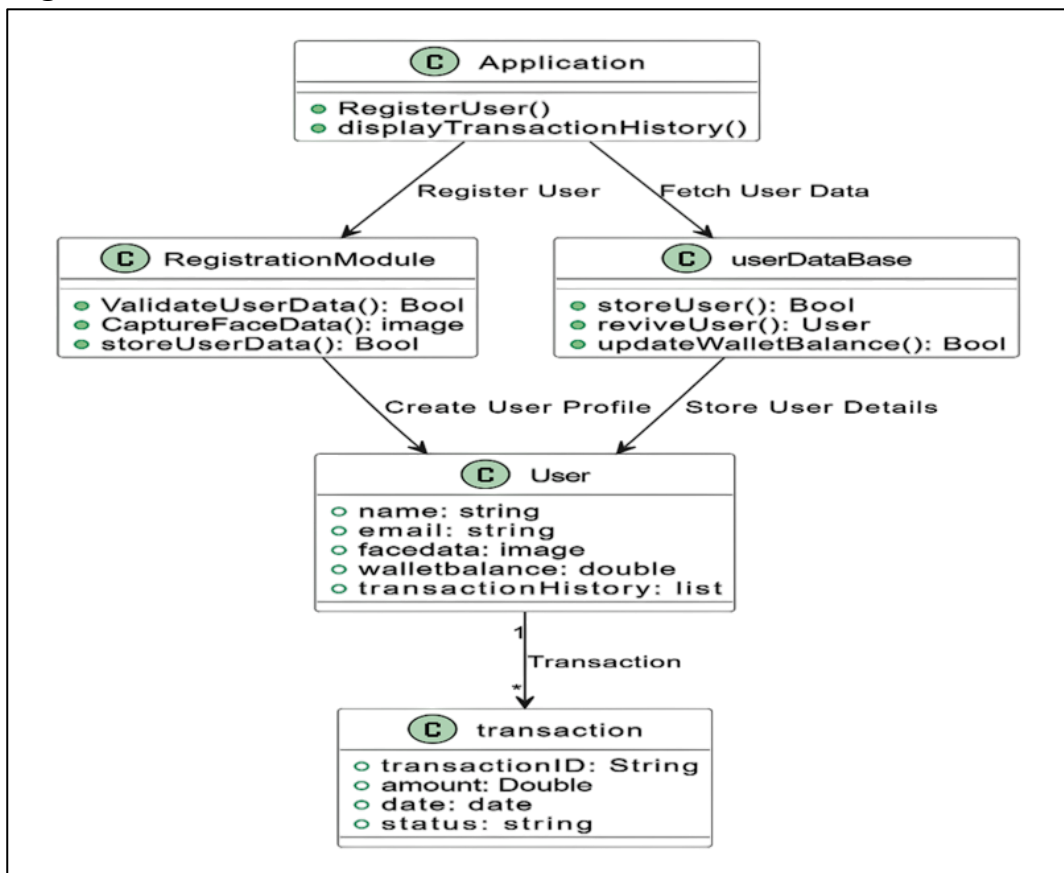
### 4.3 Modules

- User dashboard Module
- Scanner payment Module
- Shop dashboard Module

#### 4.3.1 User Dashboard Module

This module provides a front-end interface for users to interact with the system via a mobile application.

#### Class Diagram

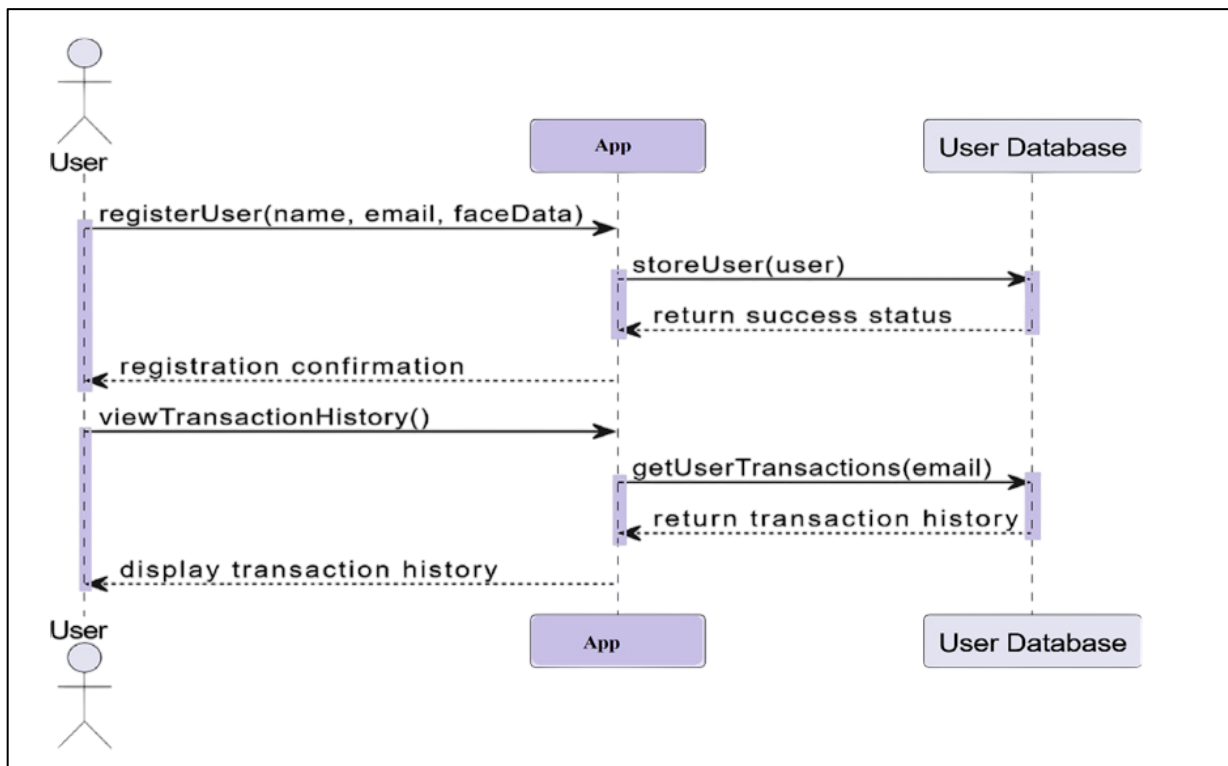


*Fig 4.3 Class Diagram for User dashboard Module*

## Description

The provided class diagram represents the User Interface Module of a face recognition payment system. It consists of multiple interconnected classes, each handling specific responsibilities within the system. The Mobile App class serves as the primary interface for users, enabling account registration and transaction history retrieval. It interacts with the Registration Module, which is responsible for validating user data, capturing facial images, and storing user details. The User Database class manages user-related operations, including storing user details, retrieving user information, and updating wallet balances. The User class represents a registered individual with attributes like name, email, facial data, wallet balance, and a list of transactions. Each User can have multiple Transaction records, where each Transaction includes details such as transaction ID, amount, date, and status. The relationships among these classes define the flow of data, ensuring proper user authentication, data storage, and payment processing functionalities. The multiplicity annotations indicate that a user can have multiple transactions, while the system can store multiple users in the database, ensuring scalability.

## Sequence Diagram

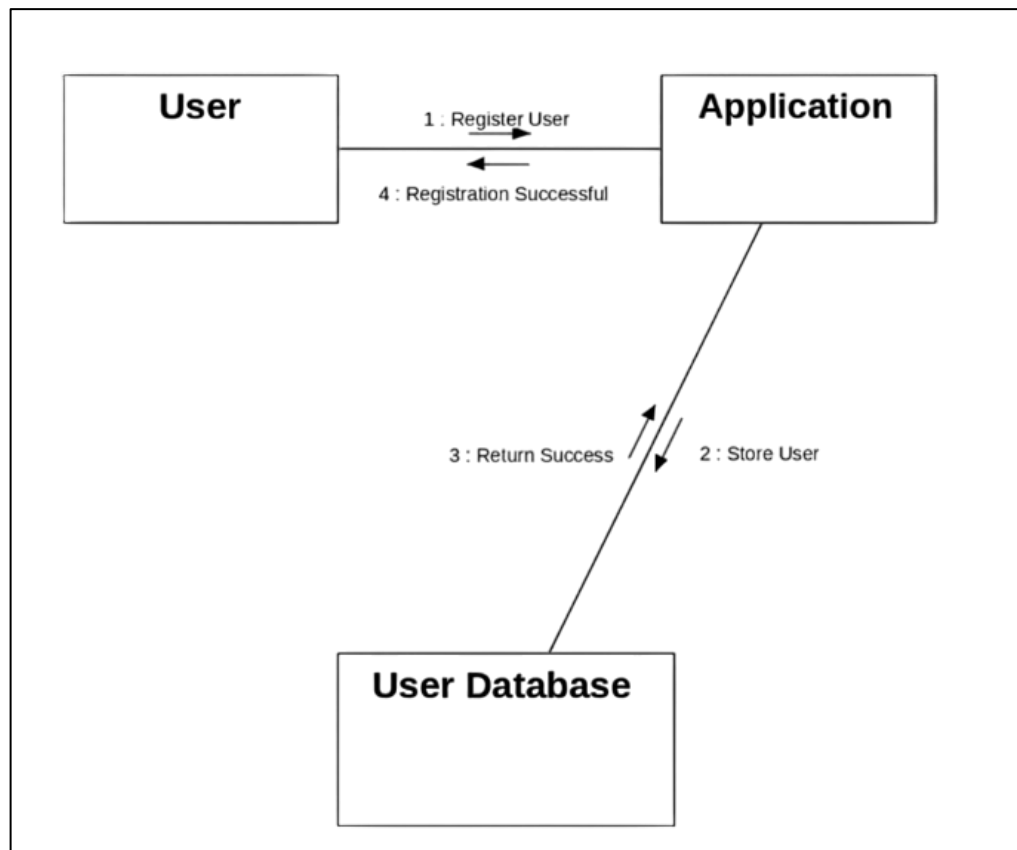


*Fig 4.4 Sequence Diagram for User dashboard Module*

## Description

The illustrates a UML sequence diagram showcasing two primary interactions between a user, a mobile application (App), and a user database. The first interaction represents the user registration process, where the user submits their name, email, and facial data to the app through the registerUser() method. The app then forwards this information to the user database via the storeUser() call. Upon successful storage, the database returns a confirmation status, which the app relays back to the user as a registration confirmation. The second interaction depicts the process of viewing transaction history. The user initiates this by calling viewTransactionHistory() on the app, which in turn requests transaction data from the user database using the getUserTransactions(email) method. The database responds with the transaction history, which is then displayed to the user. This diagram effectively visualizes the sequential flow of data and operations involved in user registration and transaction history retrieval within the system.

## Collaboration Diagram



*Fig 4.5 Collaboration Diagram for User dashboard Module*

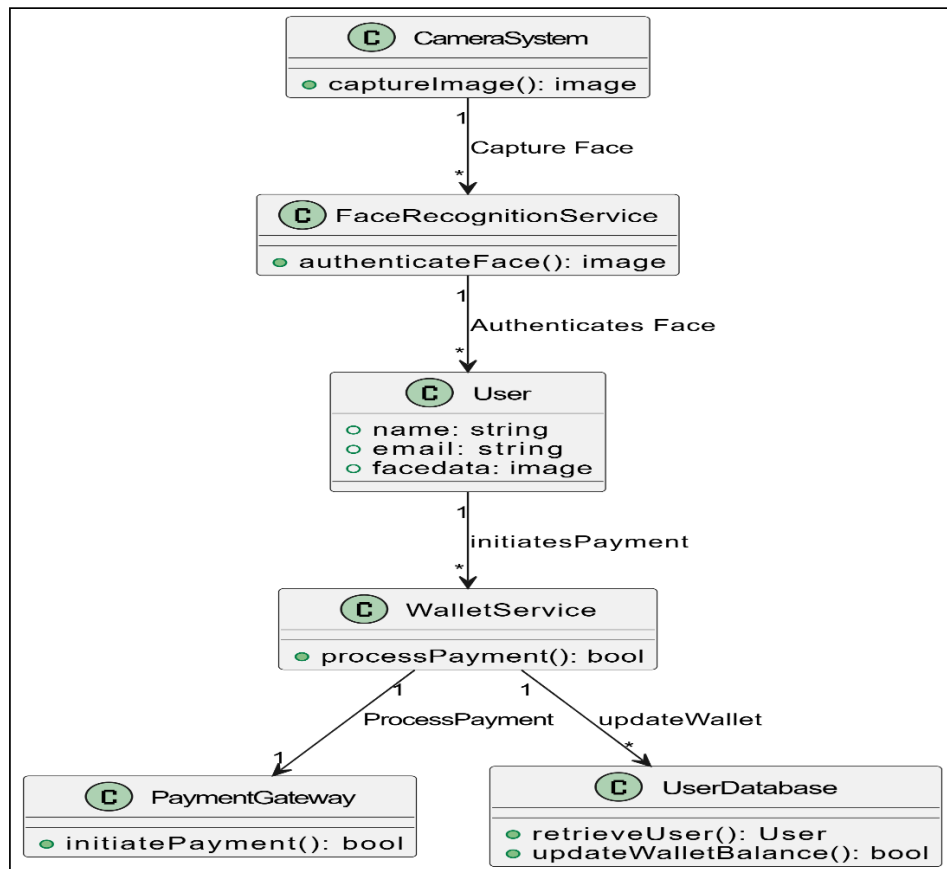
## Description

The image illustrates a UML collaboration (communication) diagram that depicts the user registration process in a system involving three components: the User, the Application, and the User Database. The sequence begins with the User sending a "Register User" request to the Application. Upon receiving the request, the Application forwards the user data to the User Database via the "Store User" message. The User Database processes this data and sends back a "Return Success" response to the Application to indicate that the user information has been stored successfully. Finally, the Application notifies the User with a "Registration Successful" message, completing the process. This diagram effectively demonstrates the interactions and message flow among system components during user registration.

### 4.3.2 Scanner Payment Module

This module is responsible for face authentication during payments.

#### Class Diagram

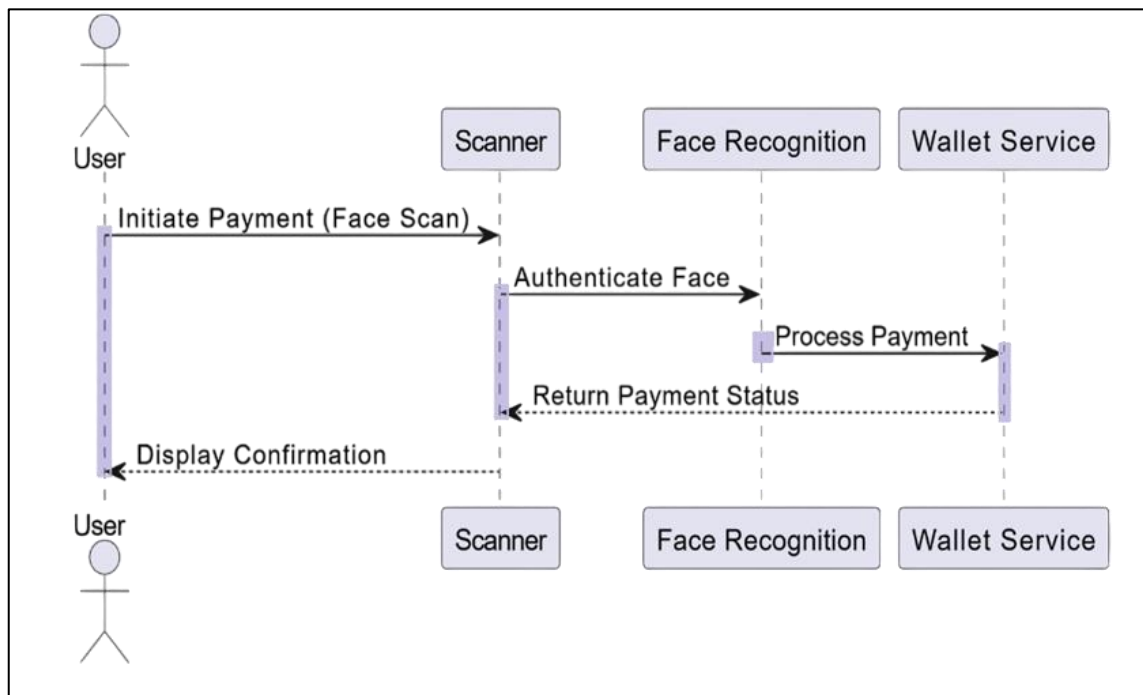


*Fig 4.6 Class Diagram for Scanner Payment Module*

## Description

The Face Recognition Module Class Diagram illustrates the interaction between various components involved in the face recognition-based payment system. The Camera System captures the user's facial image and sends it to the Facial Recognition Service, which authenticates the face. If authentication is successful, the Wallet Service processes the payment by verifying the user and the transaction amount. The Payment Gateway initiates the payment, while the Transaction Database logs the transaction details. The User Database is responsible for storing and retrieving user data, as well as updating wallet balances. The User entity holds personal details, including name, email, and face data, and interacts with both the Facial Recognition Service for authentication and the Transaction Database for logging transactions. The diagram effectively represents the workflow of facial recognition and payment processing while maintaining secure user authentication and transaction tracking.

## Sequence Diagram



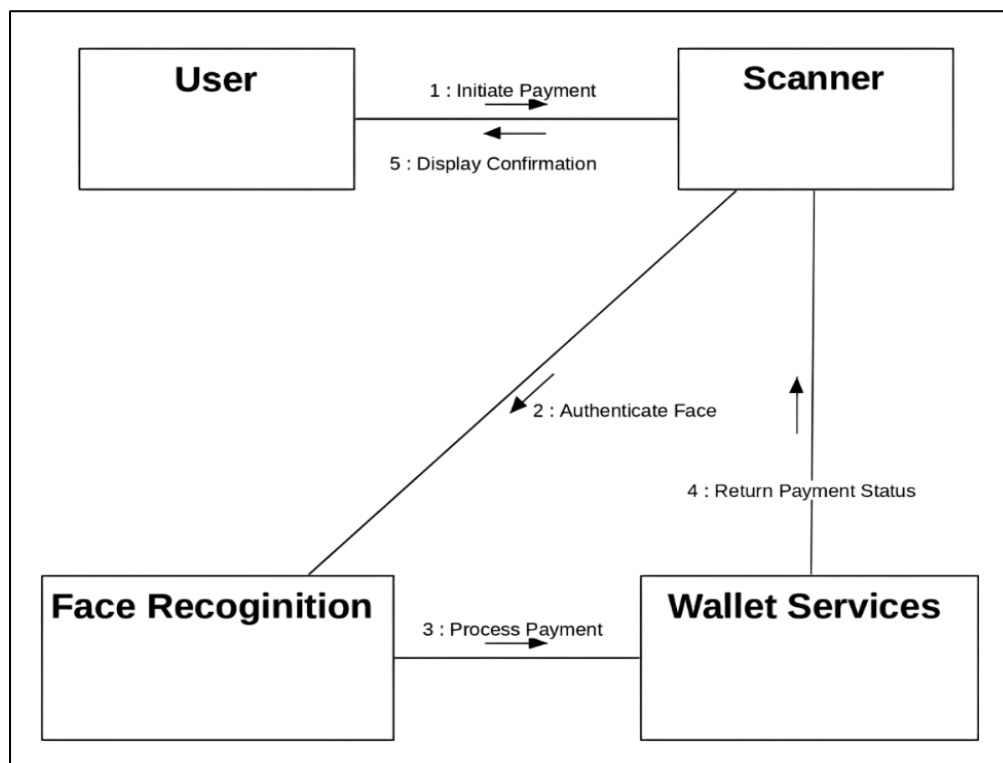
*Fig 4.7 Sequence Diagram Scanner Payment Module*

## Description

The image represents a UML sequence diagram illustrating the process of initiating and completing a payment using facial recognition technology. The sequence begins with the User initiating a payment through a Scanner, which triggers a face scan. The Scanner then sends the

facial data to the Face Recognition module for authentication. Once the user is successfully authenticated, the Face Recognition component forwards the request to the Wallet Service to process the payment. Upon completion, the Wallet Service returns the payment status back to the Face Recognition module, which in turn relays it to the Scanner. Finally, the Scanner provides feedback to the User by displaying the payment confirmation. This diagram effectively demonstrates the interaction between different components in a face recognition-based payment system.

### Collaboration Diagram



*Fig 4.8 Collaboration Diagram for Scanner Payment Module*

### Description

The diagram is a UML communication view of a face-recognition payment flow involving four roles: User, Scanner, Face Recognition, and Wallet Services. First, the User interacts with the Scanner to initiate payment (1: Initiate Payment). The Scanner then forwards the captured facial data to the Face Recognition component to authenticate the face (2: Authenticate Face). Once authenticated, Face Recognition sends a payment request to Wallet Services (3: Process Payment). Wallet Services processes the transaction and returns the payment status back to the Scanner (4: Return Payment Status). The Scanner then displays the payment confirmation to the User (5: Display Confirmation).

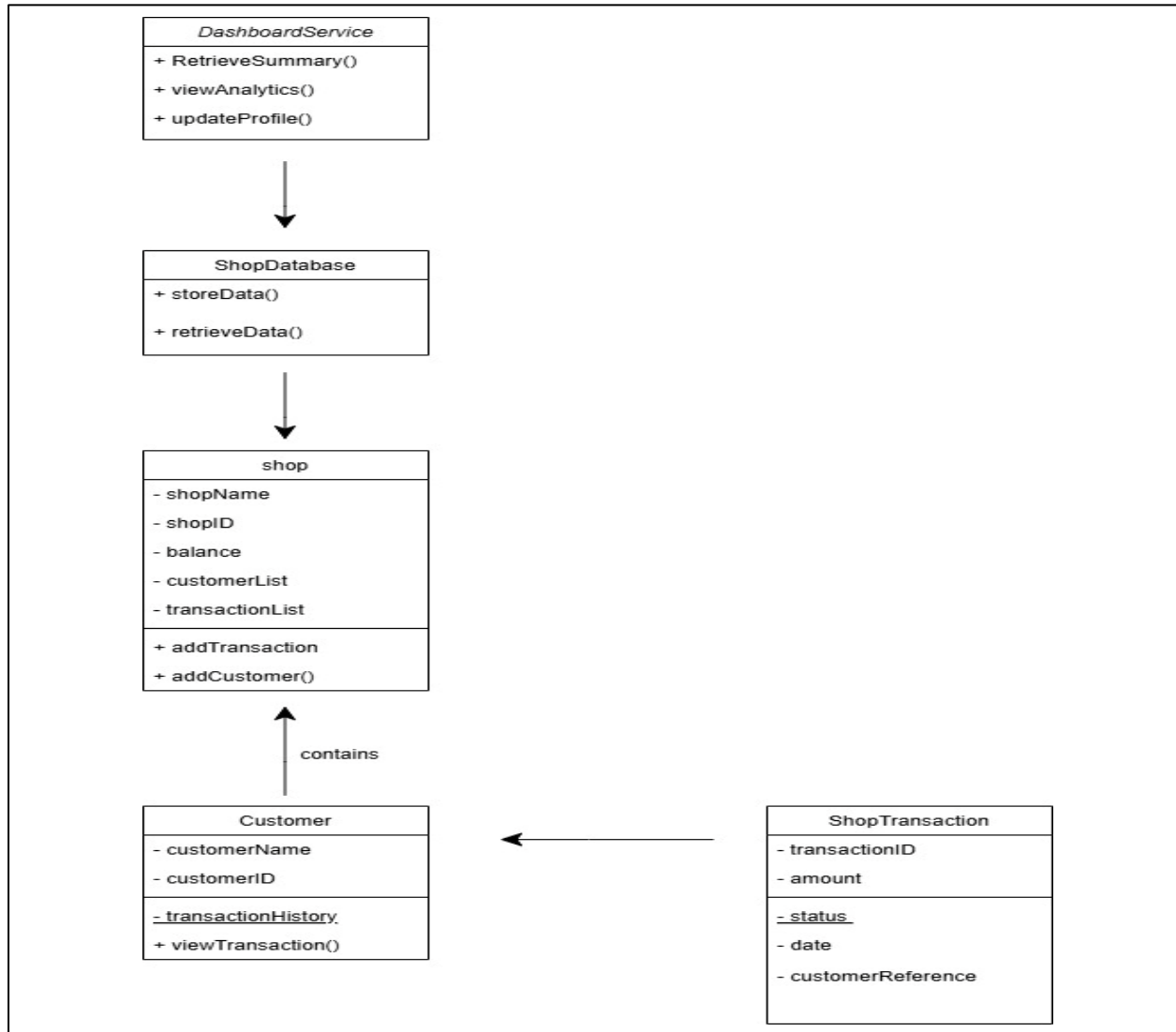


Return Payment Status). Finally, the Scanner displays the confirmation or error message to the User (5: Display Confirmation). This numbered sequence clearly shows how each component collaborates to complete a secure, face-based payment.

### 4.3.3 Shop Dashboard Module

This module provides merchants with real-time insights into their shop's performance, including transaction summaries, customer activity, and wallet balance updates

#### Class Diagram



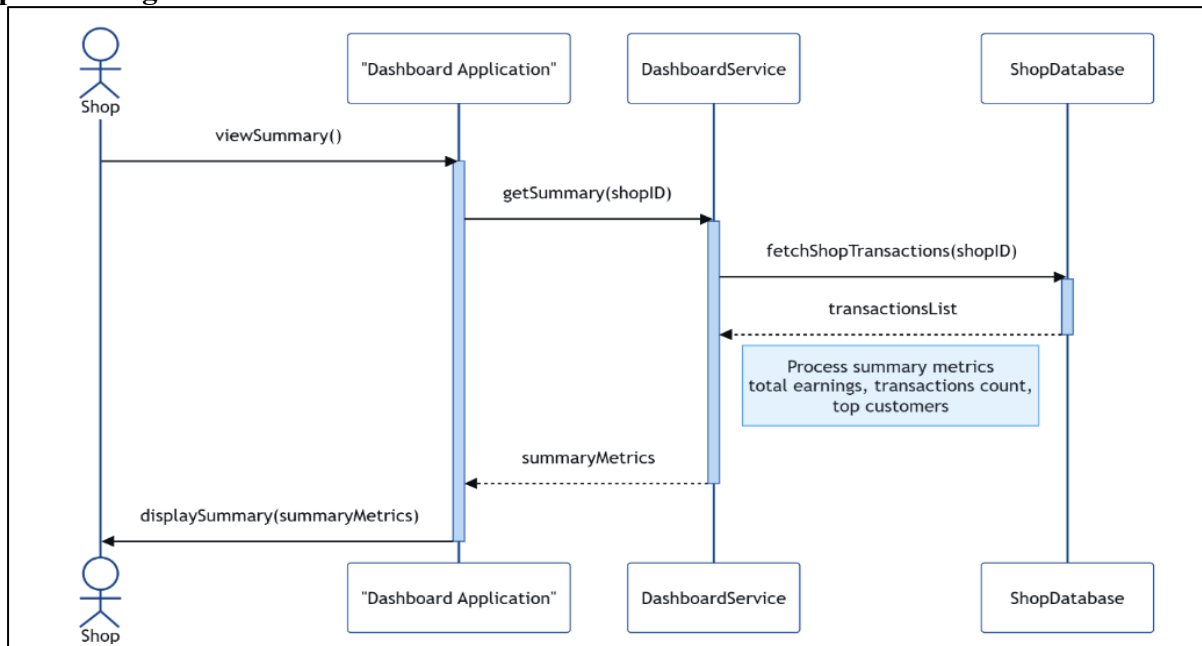
*Fig 4.9 Class Diagram for Shop Dashboard Module*

#### Description

The class diagram illustrates the Backend Services Module of a face recognition payment system. It consists of five primary classes: Backend Service, User Database, Transaction Database,

User, and Transaction. The Backend Service class manages operations such as processing payments, fetching transaction history, and updating wallet balances. It interacts with both the User Database and Transaction Database, each of which handles user and transaction data storage, respectively. The User Database class stores multiple users, while the Transaction Database maintains records of transactions. The User class represents individuals in the system with attributes such as name, email, face data, wallet balance, and transaction history. Each User can have multiple Transactions, represented by the Transaction class, which includes transaction details like transaction ID, amount, date, and status. The relationships are well-defined, showing one-to-many associations where a user can have multiple transactions, and databases can store multiple user and transaction records. This diagram provides a clear structural overview of the backend services, ensuring efficient data management and processing within the payment system.

### Sequence Diagram



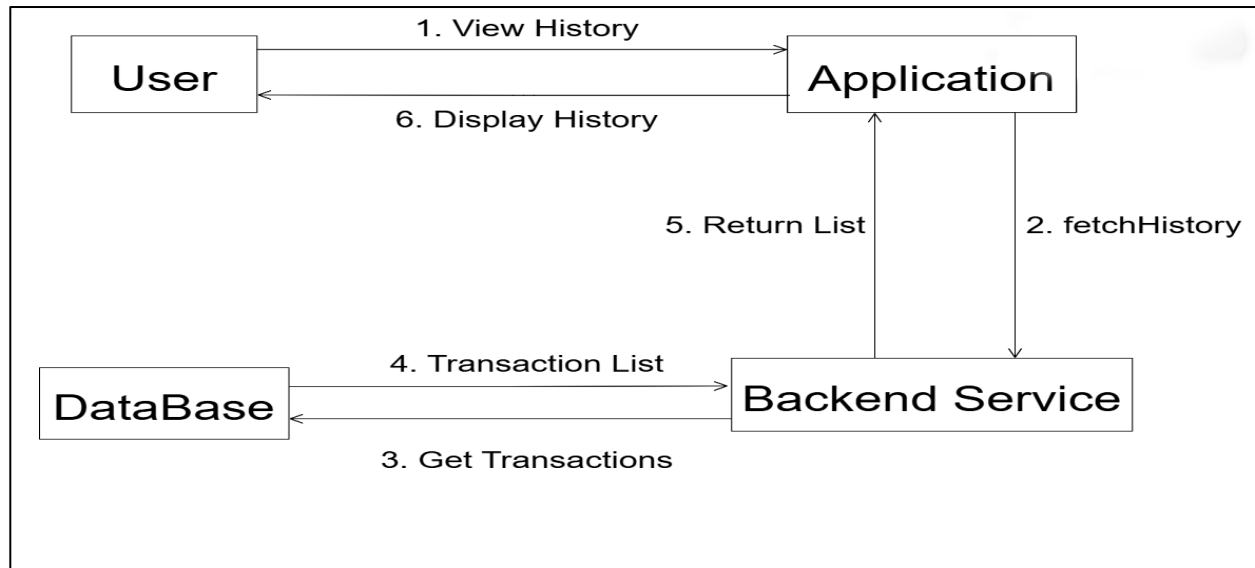
*Fig 4.10 Sequence Diagram for Shop Dashboard Module*

### Description

The diagram is a UML sequence chart illustrating how a user's transaction history is retrieved and displayed. It begins with the User invoking `viewHistory()` on the Application, which in turn calls `fetchHistory(user)` on the Backend Service. The Backend Service then queries the Database by sending `getTransactions(user)`, and the database responds with the user's transaction list. That list is returned back through the Backend Service to the Application via return list, and

finally the Application invokes `displayHistory()` to present the transaction history to the User. This flow clearly shows the step-by-step handoff of the user's request from the front-end to the back-end and data layer, culminating in the display of the retrieved records.

### Collaboration Diagram



*Fig 4.11 Collaboration Diagram for Shop Dashboard Module*

### Description

The image presents a flow diagram representing the process of retrieving a user's transaction history in a digital system. The sequence begins when the User initiates the process by sending a "View History" request to the Application. In response, the Application calls the Backend Service using a fetch History request. The Backend Service then communicates with the Database, sending a `getTransactions` request to retrieve the user's transaction data. The Database responds with the Transaction List, which is sent back to the Backend Service. The backend then returns this list to the Application, which finally sends the data back to the User in a Display History response. This diagram effectively outlines a standard layered architecture for handling transaction history retrieval, emphasizing the interactions between user interface, backend logic, and data storage components.

## CHAPTER V

### SYSTEM IMPLEMENTATION

#### 5.1 Sample Code

##### Face Pay User Website(app.py)

```
from flask import Flask, render_template, request, redirect, url_for
import sqlite3, os, cv2, numpy as np
app = Flask(__name__)
DATABASE = 'database.db'
os.makedirs('face_data', exist_ok=True)
def init_db():
    with sqlite3.connect(DATABASE) as conn:
        conn.execute("""CREATE TABLE IF NOT EXISTS users (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            name TEXT,
            email TEXT,
            card_number TEXT,
            expiry TEXT,
            ccv TEXT,
            password TEXT,
            wallet INTEGER DEFAULT 100
        )""")
@app.route('/')
def home():
    return render_template('index.html')
@app.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        name = request.form['name']
        email = request.form['email']
        card = request.form['card']
        expiry = request.form['expiry']
```

```

ccv = request.form['ccv']
password = request.form['password']
with sqlite3.connect(DATABASE) as conn:
    cur = conn.cursor()
    cur.execute("INSERT INTO users (name, email, card_number, expiry, ccv, password)
VALUES (?, ?, ?, ?, ?, ?)",
                (name, email, card, expiry, ccv, password))
    user_id = cur.lastrowid
    return redirect(url_for('capture_face', user_id=user_id))
return render_template('register.html')
@app.route('/capture/<int:user_id>')
def capture_face(user_id):
    cap = cv2.VideoCapture(0)
    detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    count = 0
    while True:
        ret, frame = cap.read()
        if not ret:
            break
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        faces = detector.detectMultiScale(gray, 1.3, 5)
        for (x, y, w, h) in faces:
            count += 1
            face_img = gray[y:y+h, x:x+w]
            cv2.imwrite(f'face_data/user_{user_id}_{count}.jpg', face_img)
            cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 0), 2)
            if count >= 20:
                break
    cv2.imshow('Capturing Face - Press Q to Quit', frame)
    if count >= 20 or cv2.waitKey(1) & 0xFF == ord('q'):
        break

```

```

cap.release()
cv2.destroyAllWindows()
if count >= 1:
    return redirect(url_for('train_model'))
else:
    return "<h3>Face not captured. Please try again.</h3>"
@app.route('/train')
def train_model():
    recognizer = cv2.face.LBPHFaceRecognizer_create()
    detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    face_samples = []
    ids = []
    for image_name in os.listdir("face_data"):
        if image_name.endswith('.jpg'):
            path = os.path.join("face_data", image_name)
            gray_img = cv2.imread(path, cv2.IMREAD_GRAYSCALE)
            try:
                user_id = int(image_name.split("_")[1])
                face_samples.append(gray_img)
                ids.append(user_id)
            except ValueError:
                continue
    if len(face_samples) < 1:
        return "<h3>No face data to train. Please register and capture faces first.</h3>"
    recognizer.train(face_samples, np.array(ids))
    recognizer.save('trainer.yml')
    return "<h3>Training completed! You can now login.</h3><a href='/login'>Login</a>"
@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        recognizer = cv2.face.LBPHFaceRecognizer_create()

```



```

if not os.path.exists('trainer.yml'):
    return "<h3>No trained data found. Please register and train first.</h3>"
recognizer.read('trainer.yml')
detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
cap = cv2.VideoCapture(0)
authenticated = False
user_id = None
while True:
    ret, frame = cap.read()
    if not ret:
        break
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = detector.detectMultiScale(gray, 1.3, 5)
    for (x, y, w, h) in faces:
        face_img = gray[y:y+h, x:x+w]
        label, confidence = recognizer.predict(face_img)
        if confidence < 70:
            authenticated = True
            user_id = label
        cap.release()
        cv2.destroyAllWindows()
        if authenticated:
            return redirect(url_for('dashboard', user_id=user_id))
    cv2.imshow('Login Face - Press Q to Quit', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
    cap.release()
    cv2.destroyAllWindows()
    return "<h3>Face not recognized. Try again.</h3>"
return render_template('login.html')
@app.route('/dashboard/<int:user_id>')

```

```

def dashboard(user_id):
    with sqlite3.connect(DATABASE) as conn:
        user = conn.execute("SELECT name, wallet FROM users WHERE id=?",
        (user_id,)).fetchone()
        return render_template('dashboard.html', user=user)
if __name__ == '__main__':
    init_db()
    app.run(debug=True)

```

### **Face Pay User Website (trainer\_model.py)**

```

import cv2
import numpy as np
import os
# Path setup
face_data_dir = 'face_data'
trainer_file = 'trainer.yml'
# Load face recognizer and face detector
recognizer = cv2.face.LBPHFaceRecognizer_create()
detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
faces = []
ids = []
# Loop through images in face_data/
for image_name in os.listdir(face_data_dir):
    if image_name.endswith('.jpg'):
        path = os.path.join(face_data_dir, image_name)
        gray_img = cv2.imread(path, cv2.IMREAD_GRAYSCALE)
        user_id = int(image_name.split('_')[1].split('.')[0]) # Extract ID from 'user_1.jpg'
        faces.append(gray_img)
        ids.append(user_id)
# Check if we have at least 1 face
if len(faces) == 0:
    print("[ERROR] No face data found in face_data/. Please capture faces first.")

```

```

else:
    print(f"[INFO] Training on {len(faces)} face(s)...")
    recognizer.train(faces, np.array(ids))
    recognizer.save(trainer_file)
    print(f"[SUCCESS] Model trained and saved to {trainer_file}")

```

### Face Pay User Website(capture.html)

```

<!-- capture.html (optional if needed) -->
<html>
<head><title>Capturing...</title></head>
<body>
    <h3>Please allow the camera. Capturing your face...</h3>
</body>
</html>

```

### Face Pay User Website(capture.html)

```

<!DOCTYPE html>
<html>
<head>
    <title>Dashboard</title>
    <link rel="stylesheet" href="/static/style.css">
</head>
<body>
    <div class="container">
        <h2>Welcome, {{ user[0] }}!</h2>
        <p>Your wallet points: <strong>{{ user[1] }}</strong></p>
    </div>
</body>
</html>

```

### Face Pay User Website(index.html)

```

<!DOCTYPE html>
<html lang="en">
<head>

```

```
<meta charset="UTF-8">
<title>FacePay System</title>
<style>
  body {
    background: linear-gradient(120deg, #89f7fe 0%, #66a6ff 100%);
    font-family: Arial, sans-serif;
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    height: 100vh;
  }
  h1 {
    color: #ffffff;
  }
  .button {
    padding: 15px 30px;
    margin: 10px;
    font-size: 18px;
    background-color: #ffffff;
    border: none;
    border-radius: 8px;
    cursor: pointer;
    transition: 0.3s;
  }
  .button:hover {
    background-color: #e0e0e0;
  }
</style>
</head>
<body>
```

```
<h1>Welcome to FacePay</h1>
<button class="button" onclick="window.location.href='/register'">Register</button>
<button class="button" onclick="window.location.href='/login'">Login</button>
</body>
</html>
```

### **Face Pay User Website(login.html)**

```
<!DOCTYPE html>
<html>
<head>
  <title>Login</title>
  <link rel="stylesheet" href="/static/style.css">
</head>
<body>
  <div class="container">
    <h2>Login with Face</h2>
    <form method="post">
      <button type="submit">Start Face Recognition</button>
    </form>
  </div>
</body>
</html>
```

### **Face Pay User Website(register.html)**

```
<!DOCTYPE html>
<html>
<head>
  <title>Register</title>
  <link rel="stylesheet" href="/static/style.css">
</head>
<body>
  <div class="container">
    <h2>Register</h2>
```

```

<form method="post">
  <input name="name" placeholder="Name" required><br>
  <input name="email" placeholder="Email" required><br>
  <input name="card" placeholder="Card Number" required><br>
  <input name="expiry" placeholder="Expiry Date" required><br>
  <input name="ccv" placeholder="CCV" required><br>
  <input name="password" placeholder="Password" type="password" required><br>
  <button type="submit">Register & Capture Face</button>
</form>
</div>
</body>
</html>

```

### Face Pay User Website(style.css)

```

body {
  background: linear-gradient(120deg, #89f7fe, #66a6ff);
  font-family: 'Segoe UI', sans-serif;
  color: #333;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
}
.container {
  background: #fff;
  padding: 30px;
  border-radius: 12px;
  box-shadow: 0 0 20px rgba(0,0,0,0.1);
  text-align: center;
}
input, button {
  display: block;

```

```

width: 100%;
margin: 10px 0;
padding: 12px;
border-radius: 8px;
border: 1px solid #ccc;
}
button {
background-color: #4caf50;
color: white;
cursor: pointer;
}
button:hover {
background-color: #45a049;
}

```

### **Face\_pay\_scanner(scanner\_app.py)**

```

import cv2
import sqlite3
from datetime import datetime
user_db = r"C:\Users\MUHILAN\Face_Pay\face_pay_web\database.db"
shop_db = 'shop.db'
# Initialize face recognizer
recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read(r"C:\Users\MUHILAN\Face_Pay\face_pay_web\trainer.yml")
detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
def get_user_name(user_id):
    with sqlite3.connect(user_db) as conn:
        result = conn.execute("SELECT name, wallet FROM users WHERE id=?",
        (user_id,)).fetchone()
    return result if result else (None, None)
def update_wallet_and_log(user_id, name, amount):
    with sqlite3.connect(user_db) as user_conn, sqlite3.connect(shop_db) as shop_conn:

```

```

        user_conn.execute("UPDATE users SET wallet = wallet - ? WHERE id = ?", (amount,
user_id))
        shop_conn.execute("INSERT INTO transactions (user_id, name, amount, timestamp)
VALUES (?, ?, ?, ?)",
                        (user_id, name, amount, datetime.now()))
        shop_conn.commit()
        user_conn.commit()
def run_scanner():
    cap = cv2.VideoCapture(0)
    print("[INFO] Scanning...")
    while True:
        ret, frame = cap.read()
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        faces = detector.detectMultiScale(gray, 1.3, 5)
        for (x, y, w, h) in faces:
            face_img = gray[y:y+h, x:x+w]
            label, confidence = recognizer.predict(face_img)
            if confidence < 70:
                name, wallet = get_user_name(label)
                if name:
                    cap.release()
                    cv2.destroyAllWindows()
                    print(f"Welcome {name}, your wallet balance: {wallet} points")
                    amount = int(input("Enter wallet points to transfer: "))
                    if amount > wallet:
                        print("Not enough wallet points!")
                    else:
                        update_wallet_and_log(label, name, amount)
                        print(f"{amount} points transferred.")
                    return
            else:

```



```

        print("Face not recognized.")
    cv2.imshow("Scanner", frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
    cap.release()
    cv2.destroyAllWindows()
if __name__ == '__main__':
    # Initialize shop DB if not exists
    with sqlite3.connect(shop_db) as conn:
        conn.execute("""CREATE TABLE IF NOT EXISTS transactions (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            user_id INTEGER,
            name TEXT,
            amount INTEGER,
            timestamp TEXT
        )""")
    run_scanner()
shop_website(shop_app.py)
from flask import Flask, render_template
import sqlite3
app = Flask(__name__)
DB = r"C:\Users\MUHILAN\Face_Pay\face_pay_scanner\shop.db"
@app.route('/')
def transactions():
    with sqlite3.connect(DB) as conn:
        rows = conn.execute("SELECT * FROM transactions ORDER BY timestamp
DESC").fetchall()
    return render_template('transactions.html', transactions=rows)
if __name__ == '__main__':
    app.run(debug=True)
shop_website(transactions.html)

```

```

<!DOCTYPE html>
<html>
<head>
  <title>Shop Transactions</title>
  <style>
    table { width: 80%; margin: auto; border-collapse: collapse; }
    th, td { padding: 10px; border: 1px solid #ddd; text-align: center; }
    th { background-color: #4CAF50; color: white; }
  </style>
</head>
<body>
  <h2 style="text-align:center;">Transaction History</h2>
  <table>
    <tr>
      <th>ID</th>
      <th>User ID</th>
      <th>Name</th>
      <th>Points Transferred</th>
      <th>Timestamp</th>
    </tr>
    {% for t in transactions %}
    <tr>
      <td>{{ t[0] }}</td>
      <td>{{ t[1] }}</td>
      <td>{{ t[2] }}</td>
      <td>{{ t[3] }}</td>
      <td>{{ t[4] }}</td>
    </tr>
    {% endfor %}
  </table>
</body>

```

</html>

## 5.2 Steps for Execution

### Part 1: User Web App

1. Open terminal and go to face\_pay\_web directory.
2. Make sure database.db exists.
3. Run the app:  

```
python app.py
```
4. Visit <http://127.0.0.1:5000>
5. Register a new user:
6. Fill in the form → it captures your face → stores it in face\_data/.
7. After capturing, it trains and generates trainer.yml.
8. You can now log in using face recognition.

### Part 2: Desktop Scanner App

1. Ensure trainer.yml and both databases exist.
2. Open terminal in face\_pay\_scanner folder.
3. Run the app:  

```
python scanner_app.py
```
4. The app will Scan face, Ask for wallet points.
5. Deduct points from database.db.
6. Log transaction to shop.db.

### Part 3: Shop Dashboard Website

1. Make sure shop.db contains transaction logs.
2. Run the Flask app:  

```
python app.py
```
3. Visit <http://127.0.0.1:5000>
4. You'll see all transactions (name, amount, time).

## **CHAPTER VI**

### **SYSTEM TESTING**

#### **6.1 Introduction**

The software which has been developed has to be tested in its validity. Testing is considered to be the least creating phase whole cycle of system design. In the real sense it is the phase, which helps to bring out the creativity of other phase makes it shine. Testing is the most important space in the software development activity. In software development lifecycle, the main aim of the testing process is the quality, the developed software is tested against attaining the required functionality and performance. During the testing process the software is worked with some particular test case and the output of the test cases are analyzed whether the software is working according to the expectations or not. The success of the testing process in determining the error is mostly depends upon the test case criteria, for testing any software we need to have a description of the expert behavior of the system and method of determining whether the observed behavior conformed to the expected behavior.

#### **6.2 Developing Methodologies**

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combination. Strict quality control procedures are used. The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the consideration used to develop the framework from developing the testing methodologies.

#### **6.3 Test Objectives**

- All fields entries must work properly.
- Detecting bugs as soon as feasible in any situation.
- Avoiding errors in project and product's final versions.
- Inspect to see whether the customer requirements criterion has been satisfied.
- Last but not least, the primary purpose of testing is to gauge the project and product level of quality.

#### **6.4 Types of Testing**

Since the error in the software can be injured at any stage, we have to carry out the testing process at different levels during the development. The basic levels of testing are

1. Unit Testing.

2. Functional testing.
3. System Testing.
4. Performance testing.
5. Integration Testing.
6. Acceptance Testing.

#### **6.4.1 Unit Testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produces valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, this relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

#### **6.4.2 Functional testing**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items: Valid Input: Identified classes of valid input must be accepted. Invalid Input: Identified classes of invalid input must be rejected. Functions: Identified functions must be exercised. Output: Identified classes of application outputs must be exercised. Systems Procedures: Interfacing systems or procedures must be invoked.

#### **6.4.3 System Testing**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

#### **6.4.4 Performance Testing**

The Performance test ensures that the output will be produced within the time limits, and the time taken by the system for compiling, giving response to the users and requests being sent to the system to retrieve the results.

### **6.4.5 Integration Testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that component or software application, for example, components in a software system or - one steps up - software applications at the company level - interact without error. The following are the types of integration testing.

- Top-down integration
- Bottom-up integration

#### **Top-down Integration**

Top-down integration testing is an integration testing technique used in order to simulate the behavior of the lower-level modules that are not yet integrated. Stubs are the modules that act as temporary placement for a called module and give the same output as that of the actual product.

#### **Bottom-up Integration**

Bottom-Up Integration testing is a kind of testing methodology in which the modules are tested from the bottom of control flow upwards. Bottom-Up integration testing is opposite to the Top-down integration testing. In this testing first the bottom modules of lower levels are tested and then the higher-level modules are tested. As the lower level components are started first, means all the complex modules are tested find. So, any errors in the complex modules will be solved in the early stage only.

### **6.4.6 Acceptance Testing**

Once the applications ready to be released the crucial step is user acceptance testing. In this step a group representing a cross section of end users tests the application. The user acceptance testing is done using real world scenarios perceptions relevant to the end users. User acceptance testing is often the final step before rolling out the application. usually, the end users who will be using the application test the application 50 before accepting the application. This type of testing gives the end users the confidence that the application being delivered to them meets their requirements.

### **6.4.7 Build the Test Plan**

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Testing helps to identify the possible

bugs in the individual components, so the component that has bugs can be identified and can be rectified from errors.

## **6.5 Quality Assurance**

Quality assurance defines the objectives of a project and reviews the overall activities. So that errors are connected early in the development process.

### **Levels of Quality Assurance**

Quality assurance comes in three main levels namely

- Testing
- Validation
- Certification

#### **Testing**

In system testing the common view is to eliminate program errors. This is extremely difficult and time consuming. Since designers cannot prove 100% accuracy. A successful test, then is one that finds errors.

#### **Validation**

Software validation checks that the software product satisfies or fits the intended use that is the software meets the user requirements, not as specification artefacts or as needs of those who will operate the software only but, as the needs of all the stakeholders. There are two ways to perform software validation, internal and external. During internal software validation it is assumed that the goals of the stakeholders were correctly understood and that they were expressed in the requirement artefacts precisely and comprehensively. If the software meets the requirement specification, it has been internally validated. External validation happens when it is performed by asking the stakeholders if the software meets the needs.<sup>51</sup> Different software development methodologies call for different levels of users and stakeholder involvement and feedback, so external validation occurs when all the stakeholders accept the software product and express that it satisfies their needs. Such final external validation requires the use of an acceptance test which is a dynamic test.

#### **Alpha and Beta Testing**

Alpha testing is a type of acceptance testing, performed to identify all possible issues/bugs before releasing the product to everyday users or the public. The focus of this testing is to simulate real users by using black box and white box techniques. The aim is to carry out the tasks that a

typical user might perform. Beta Testing of a product is performed by "real users of the software application in a "real environment and can be considered as a form of external User Acceptance Testing. Beta version of the software is released to a limited number of end-users of the product to obtain feedback on the product quality. Beta testing reduces product failure risks and provides increased quality of the product through custom validation.

### **6.5.1 Generic risk**

Risk is an expectation of loss, a potential problem that may or may not occur in the future. It is generally caused due to lack of information, control of Time. A possibility of suffering from loss in the software development process is called a software risk. Reduce the probability or likelihood of risk.

#### **Risk monitoring**

- A less associated with the event.
- The likelihood that the event will occur.
- The degree to which we can change the outcome.

### **Security Technologies and Policies**

The software quality assurance consists of a variety of tasks associated with seven major activities.

- Application of technical methods
- Conduct of formal technical reviews
- Software testing
- Enforcement of standards<sup>52</sup>
- Control of change
- Measurement
- Record keeping and reporting

Security is part of the software development process, is an ongoing process involving people and practices, and ensures application confidentiality, integrity, and availability. Secure software is the result of a security aware software development process where security is built in and the software is developed with security in mind. Security is most effective if planned and managed throughout every stage of Software Development Life Cycle (SDLC), especially in critical applications or those that process sensitive information. Common attributes of security testing include authentication, authorization, confidentiality, availability, integrity, non-repudiation and resilience. Security testing is essential to ensure that the system prevents



unauthorized users from accessing its resources and data. some application data is sent over the internet which travels through a series of servers and network devices. This gives ample opportunities to unscrupulous hackers. All secure systems implement security controls within the software, Hardware, systems and networks- each component or process has a layer of isolation to protect an organization's most valuable resource which is its data. There are various security controls that can be incorporated into an application's development process to ensure sound and prevent unauthorized access.

### Modules Covered

- **User Web App:** Registration, Face Capture, Login, Dashboard
- **Desktop Scanner App:** Face Authentication, Wallet Deduction, Shop Logging
- **Shop Dashboard Website:** View Transactions

### 1. Functional Testing

#### User Web App

Test Case ID	Description	Input	Expected Output	Actual Output	Status
TC-001	Register with valid data	Name, Email, Card, Expiry, CCV, Password	Redirect to face Capture	Works as expected	Working
TC-002	Register with missing fields	Blank fields	Show error or stay on form	Validated properly	Working
TC-003	Face capture after registration	Face in front of camera	Save 20 images in face_data/	Images saved	Working
TC-004	Train model	After face capture	trainer.yml created	File created	Working
TC-005	Login with registered face	Real face input	Redirect to dashboard	Works correctly	Working
TC-006	Login with unregistered face	Unknown face	Show error message	Error displayed	Working
TC-007	View wallet points on dashboard	User logged in	Shows 100 points initially	Displayed correctly	Working

## Desktop Scanner App

Test Case ID	Description	Input	Expected Output	Actual Output	Status
TC-008	Run face scan	Face of registered user	User authenticated	Success	Working
TC-009	Wallet point deduction	Enter valid number (e.g., 10)	Deduct points from database	Wallet updated	Working
TC-010	Wallet point exceeds balance	Enter value > balance	Error message shown	Handled properly	Working
TC-011	Transaction log entry	After successful scan	Entry in shop.db	Logged with name, points	Working
TC-012	Enter invalid (non-numeric) points	abc	Show validation error	Properly handled	Working

## Shopkeeper Dashboard

Test Case ID	Description	Input	Expected Output	Actual Output	Status
TC-013	View transaction log	Open dashboard URL	Show transaction table	Records visible	Working
TC-014	Check record details	Name, Points	Matches shop.db entries	Consistent	Working
TC-015	Check when no transactions	Empty DB	Show “No transactions” message	Works fine	Working

## 2. Integration Testing

Integration tests check the interaction between modules.

Test Case ID	Description	Modules Involved	Expected Result	Status
INT-001	Registration → Face capture → Training → Login	Web App	Seamless flow, success at each step	Working
INT-002	Face scanner → Deduct points → Update database	Scanner + Web DB	Wallet value decreased in database	Working

INT-003	Scanner → Log transaction → Shop view	Scanner + Shop DB + Shop Website	Transaction appears in dashboard	Working
---------	---------------------------------------	----------------------------------	----------------------------------	---------

### 3. Security Testing

Test Case ID	Description	Modules Involved	Expected Result	Status
SEC-001	Direct access to dashboard without login	URL: /dashboard/1	Should require login or show error	Working
SEC-002	SQL Injection on login fields	' OR 1=1--	Prevent access	Working
SEC-003	Password stored in plaintext	Check DB	Should ideally be hashed	Not Working

### 4. Usability Testing

Test Case ID	Description	Area	Expected Result	Status
UST-001	UI is clear and easy to use	Web	All steps are simple and navigable	Working
UST-002	Camera feedback during face capture	Web + Scanner	Shows real-time video with boxes	Working
UST-003	Shop dashboard is readable and neat	Shop Site	Table is clean and responsive	Working

## **CHAPTER VII**

### **CONCLUSION AND FUTURE ENHANCEMENT**

#### **Conclusion**

The Face Recognition Payment System provides a secure and contactless method of transaction using biometric authentication. It integrates facial recognition with wallet-based transactions, allowing users to register their facial data and use it for future payments. The system is comprised of three modules: a web platform for user registration, a desktop-based face scanner for authenticating and deducting wallet points, and a shopkeeper dashboard for viewing incoming transactions. Through this system, we demonstrate a novel way of integrating AI-based facial recognition with traditional payment systems, offering better convenience and security in retail environments. The modular design allows scalability and easier deployment across platforms.

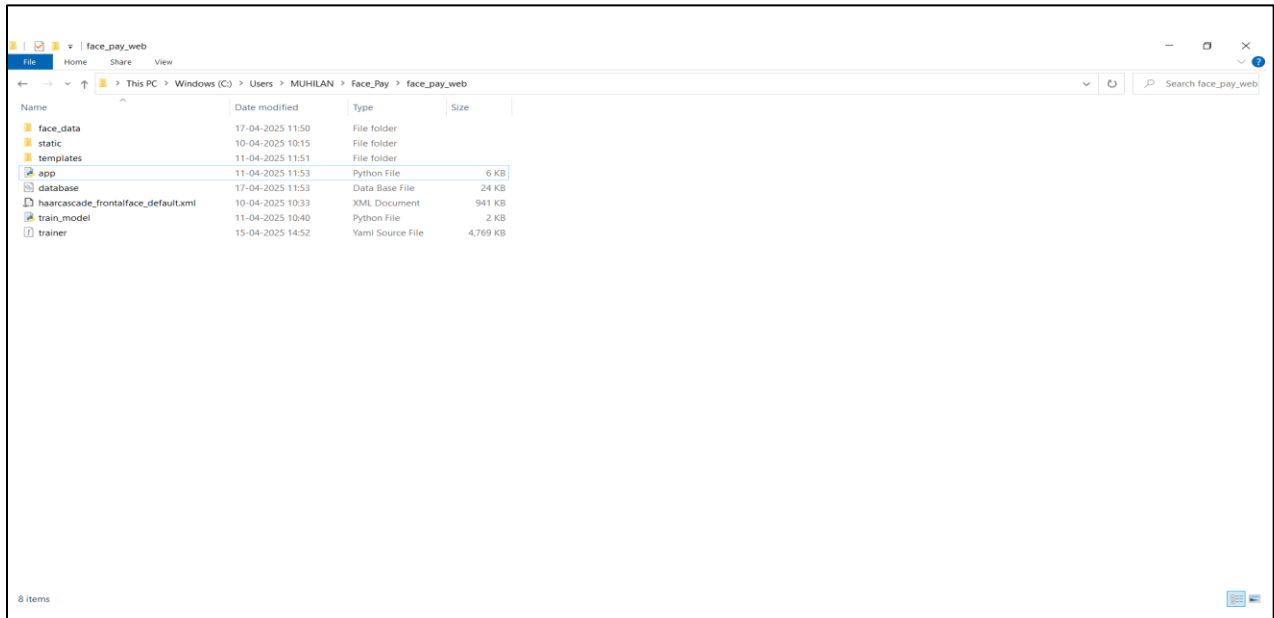
#### **Future Enhancement**

To enhance the system's capabilities and real-world applicability, several future improvements are proposed. These include implementing multi-factor authentication by combining facial recognition with OTP or PIN for added security. A mobile face scanner app will allow users to authenticate and transact on-the-go, increasing accessibility and convenience. Upgrading to advanced deep learning models like FaceNet or Dlib will improve facial recognition accuracy, even under varied lighting and environmental conditions. Integration with real payment gateways such as Stripe, Razorpay, or PayPal will enable live financial transactions and wallet top-ups using secure APIs. An admin panel for shopkeepers will offer better usability with features like transaction history, sorting, and report generation, streamlining business operations. User notifications via email or SMS, powered by tools like Firebase Cloud Messaging or Twilio, will provide instant updates on transaction status or low balance alerts. Additionally, implementing end-to-end data encryption using standards like AES-256 and TLS will enhance both data-at-rest and data-in-transit security. The system can also be expanded to support attendance tracking and access control modules, using facial biometrics for employee management. These enhancements aim to build a more secure, scalable, and intelligent facial recognition system suitable for financial and enterprise applications.

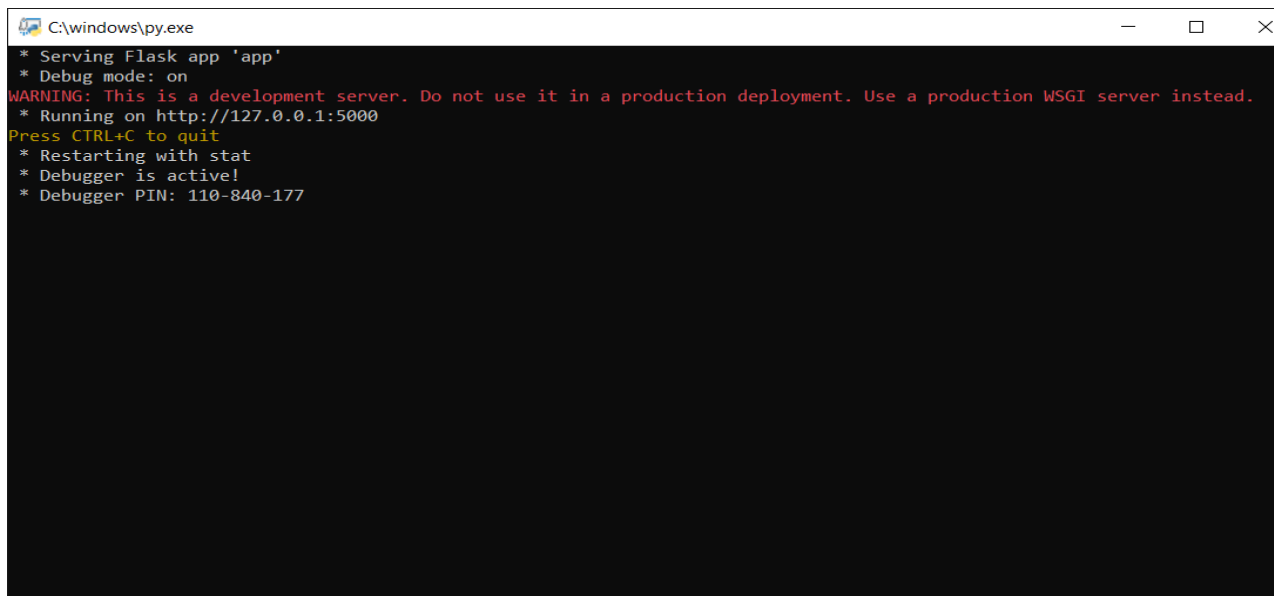
# APPENDICES

## APPENDIX I

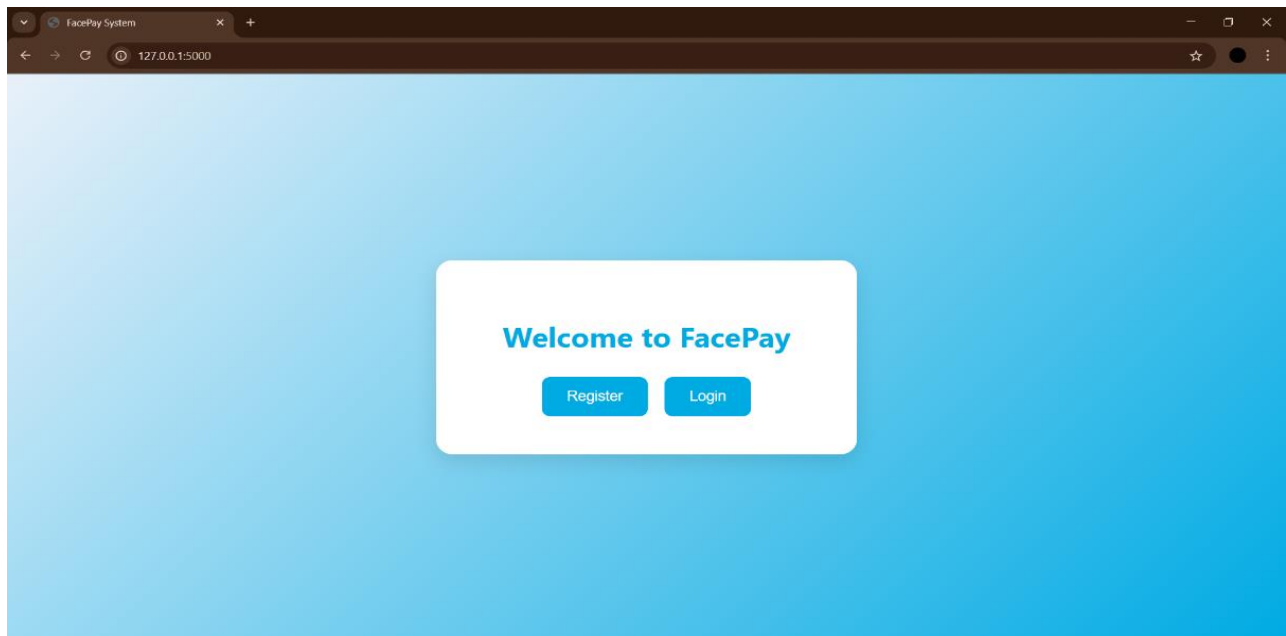
### A1.1 Screenshot



*Fig.A1.1 Opening the face pay web app*



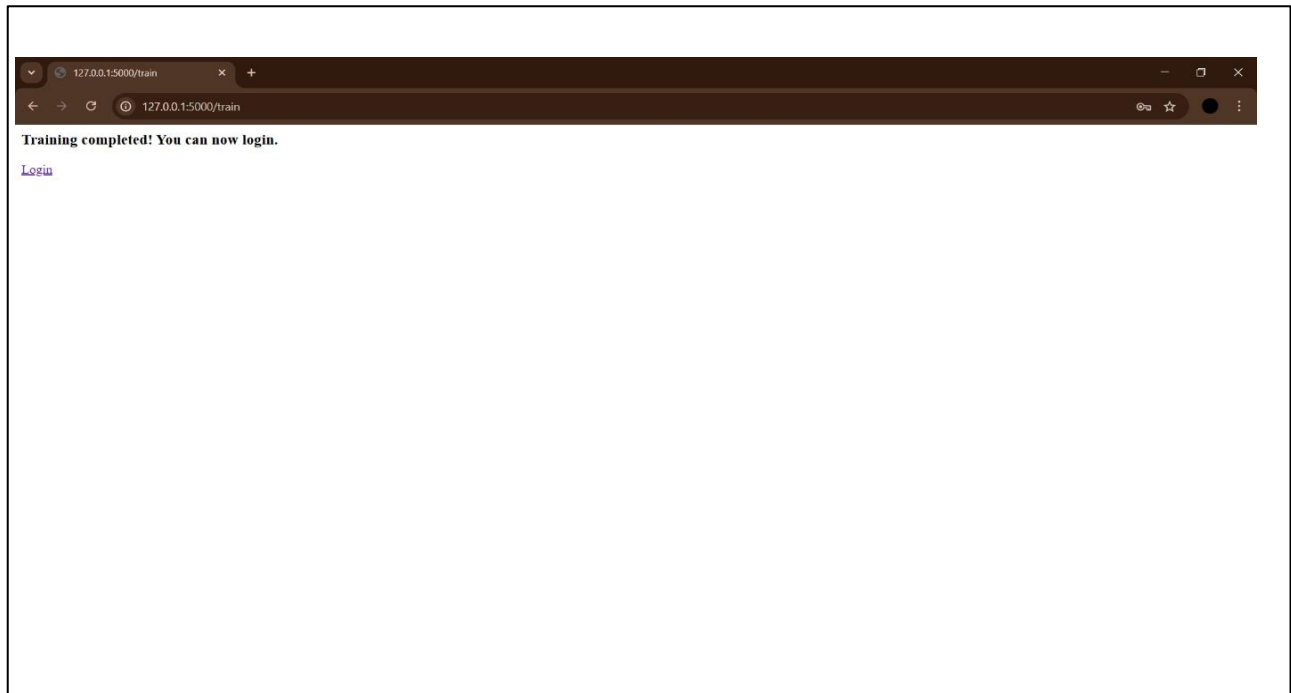
*Fig.A1.2 Running the face pay web app*



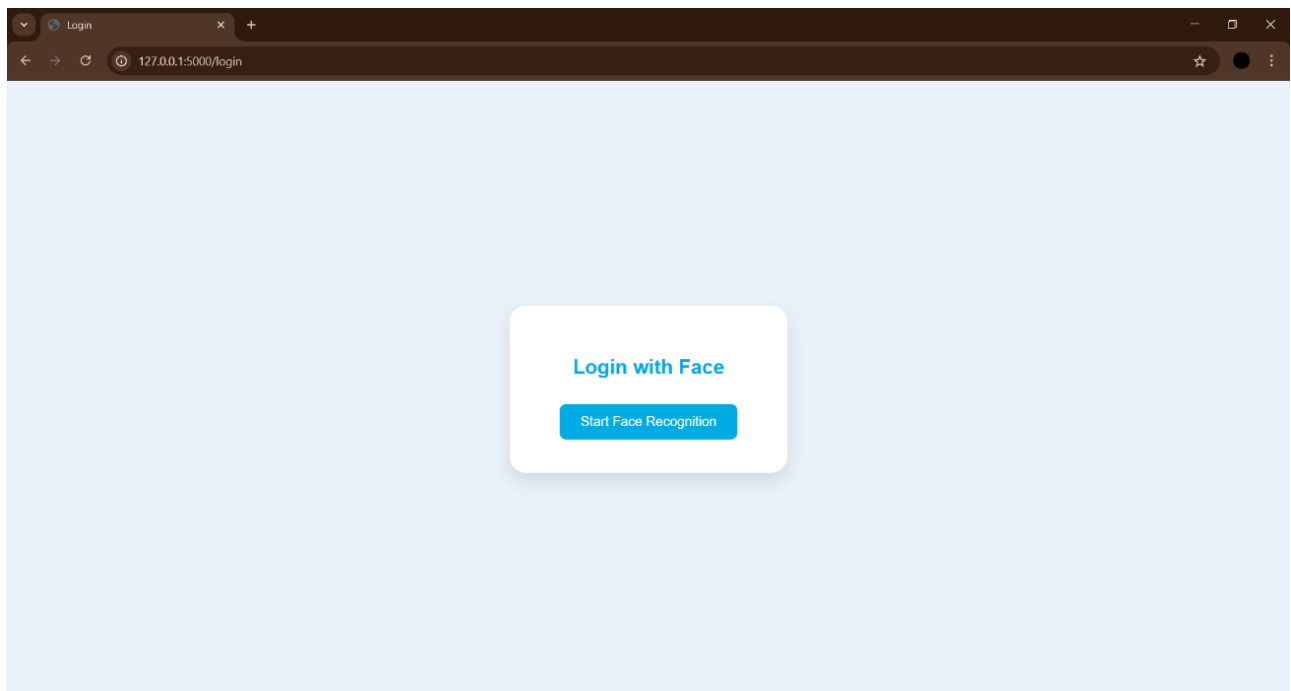
*Fig.A1.3 Interface of face pay web app*

A screenshot of a web browser displaying the FacePay registration page. The browser's address bar shows the URL "127.0.0.1:5000/register". The page has a light blue gradient background. In the center, there is a white rounded rectangle containing a registration form. At the top of the form is a circular profile picture placeholder with a cartoon character. Below the picture is the text "Create Your FacePay Account" in a bold, dark blue font. The form consists of several input fields, each with a blue icon on the right: a name field with the text "Raghul" and a person icon; an email field with the text "raghultharun23@gmail.com" and an envelope icon; a phone number field with the text "123412341234" and a phone icon; a date of birth field with the text "11/27" and a calendar icon; a PIN field with the text "333" and a lock icon; and a password field with the text "\*\*\*\*" and a key icon. Below the input fields is a blue button with white text that says "Register & Capture Face". At the bottom of the form, there is a link that says "Already have an account? Login here".

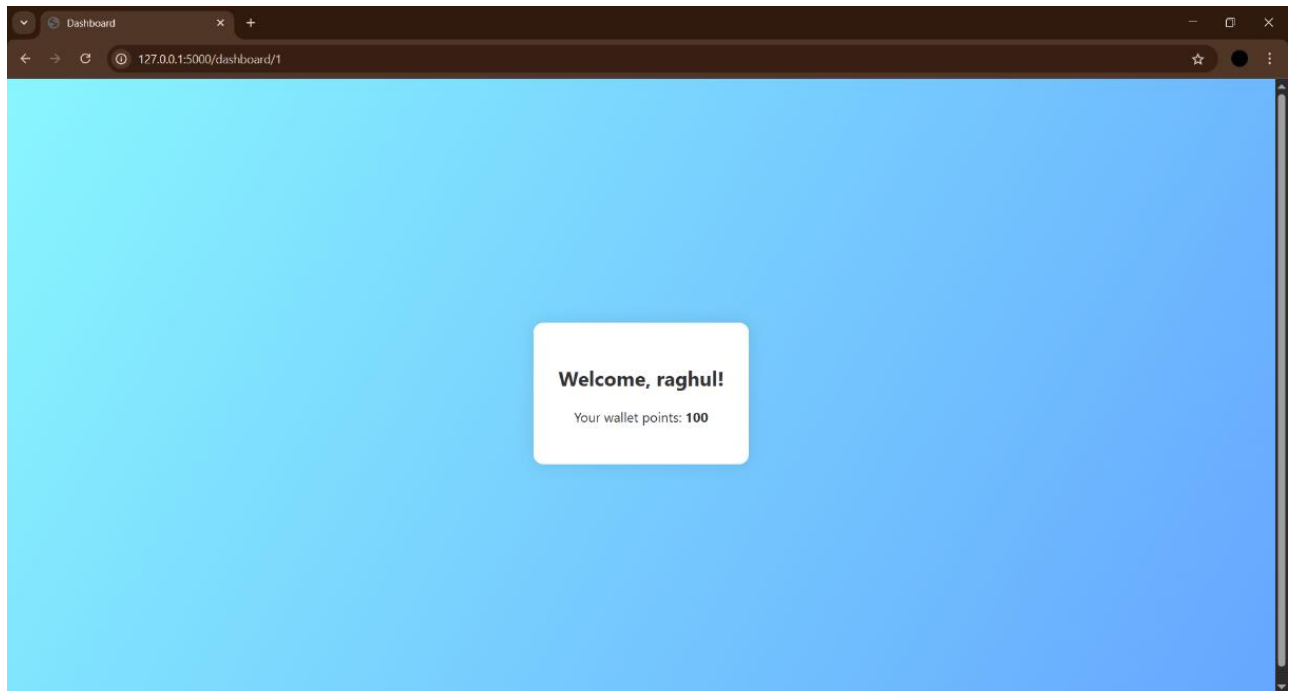
*Fig.A1.4 Registering our data*



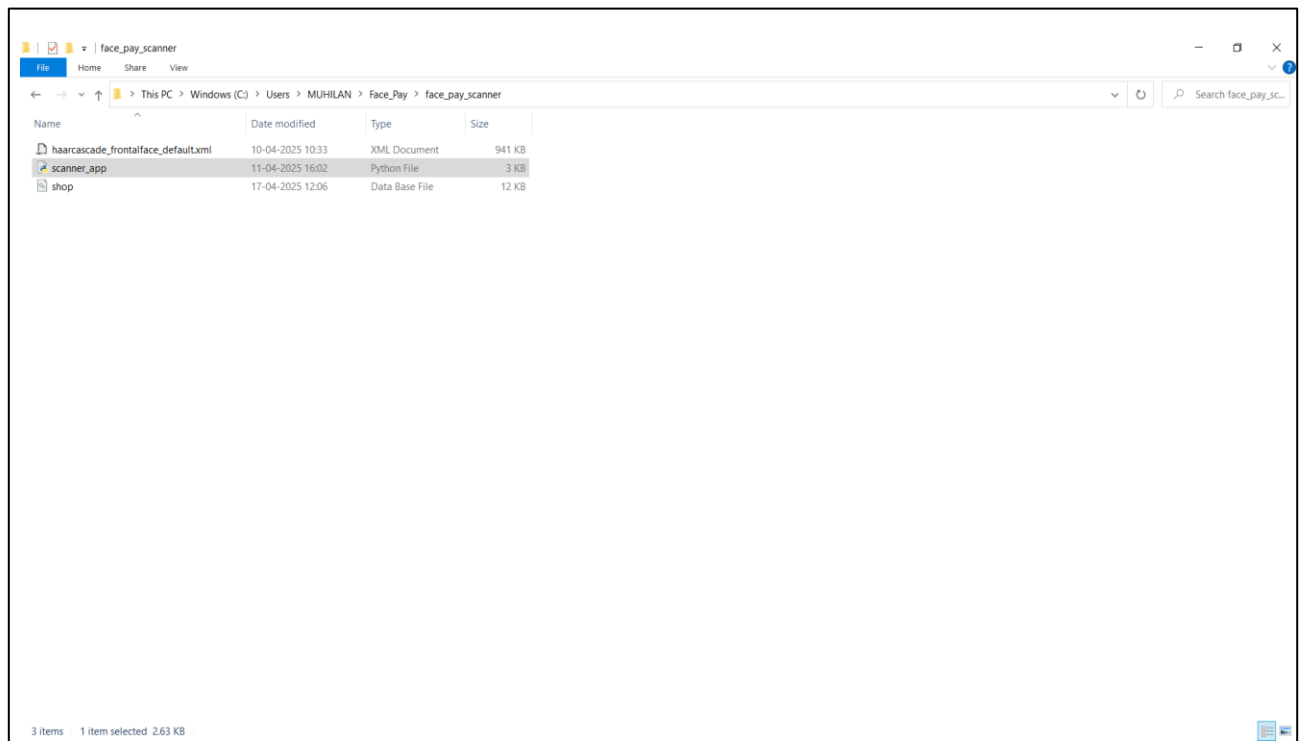
*Fig.A1.5 Face capturing and Training*



*Fig.A1.6 Login Interface*

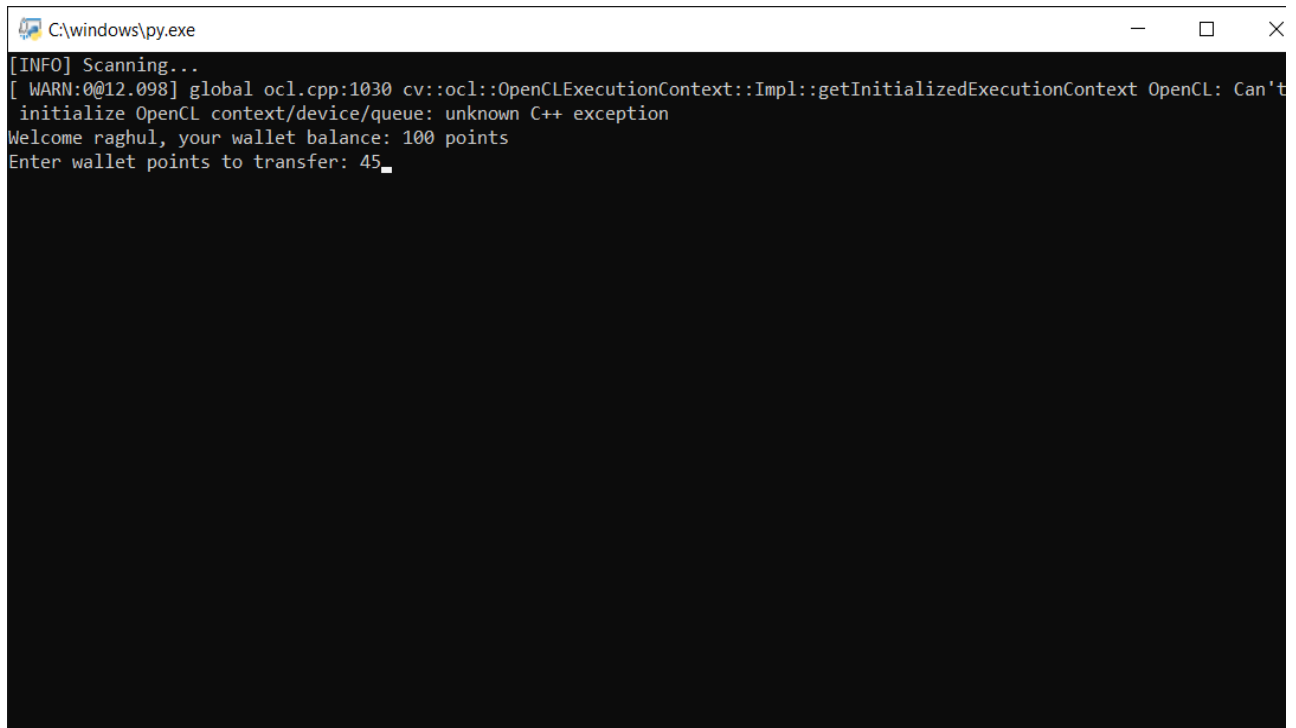


*Fig.A1.7 Login Page*



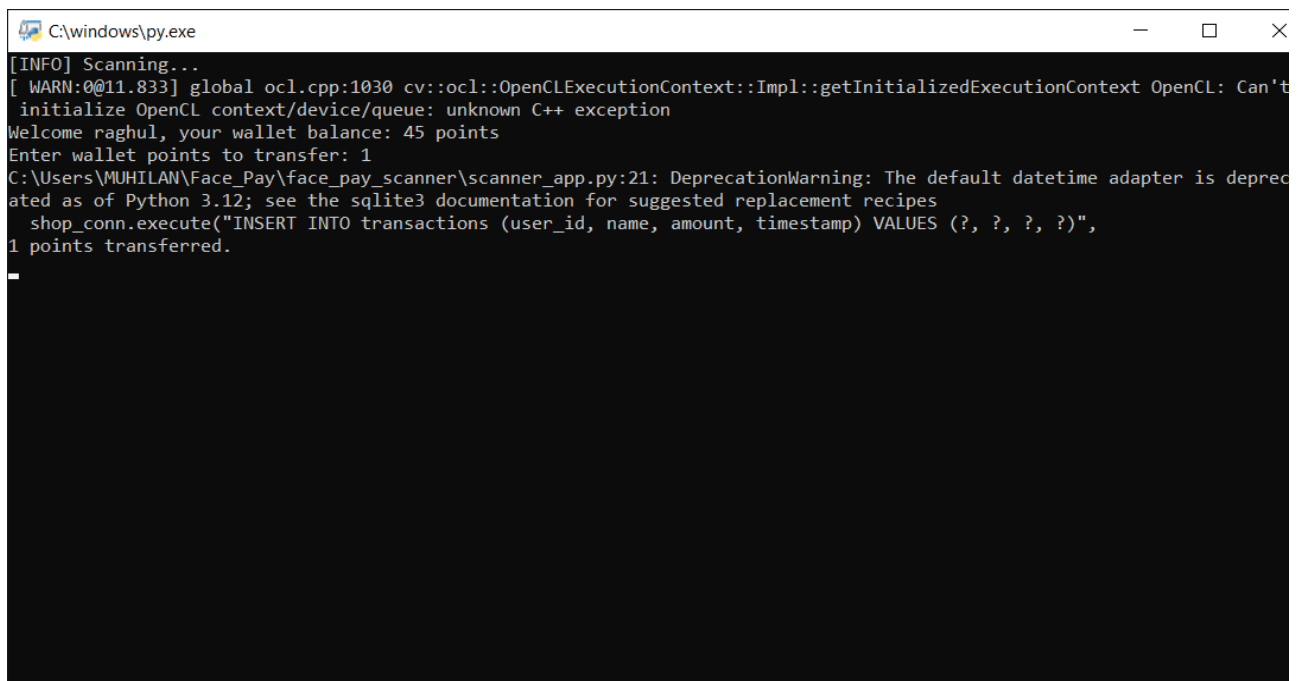
*Fig.A1.8 Opening the folder of scanner app*





```
C:\windows\py.exe
[INFO] Scanning...
[ WARN:0@12.098] global ocl.cpp:1030 cv::ocl::OpenCLExecutionContext::Impl::getInitializedExecutionContext OpenCL: Can't
initialize OpenCL context/device/queue: unknown C++ exception
Welcome raghul, your wallet balance: 100 points
Enter wallet points to transfer: 45
```

*Fig.A1.9 Face capturing and wallet transferring*



```
C:\windows\py.exe
[INFO] Scanning...
[ WARN:0@11.833] global ocl.cpp:1030 cv::ocl::OpenCLExecutionContext::Impl::getInitializedExecutionContext OpenCL: Can't
initialize OpenCL context/device/queue: unknown C++ exception
Welcome raghul, your wallet balance: 45 points
Enter wallet points to transfer: 1
C:\Users\MUHILAN\Face_Pay\face_pay_scanner\scanner_app.py:21: DeprecationWarning: The default datetime adapter is deprec
ated as of Python 3.12; see the sqlite3 documentation for suggested replacement recipes
  shop_conn.execute("INSERT INTO transactions (user_id, name, amount, timestamp) VALUES (?, ?, ?, ?)",
1 points transferred.
```

*Fig.A1.10 Wallet transferred*

The screenshot shows a web browser window with the title 'Shop Transactions'. The address bar displays '127.0.0.1:5000'. The main content area has a blue header with the text 'Retail Shop Transactions'. Below the header, there is a small icon and the text 'Retail Shop'. The main section is titled 'Transaction History' and contains a table with the following data:

ID	User ID	Name	Points Transferred	Timestamp
3	10	Raghul	1	2025-05-22 21:19:03.788961
2	10	Raghul	55	2025-05-22 21:18:38.504085

*Fig.A1.11 Shop website*

## **APPENDIX II**

### **REFERENCES AND BIBLIOGRAPHY**

#### **A2.1 References**

- [1] Facial-Recognition Payment: An Example of Chinese Consumers, Wen Kun Zhang ; Min Jung Kang, IEEE Access, Year: 2019
- [2] Secure multifactor authentication payment system using NFC, Anirudhan Adukkathayar; Gokul S Krishnan ; Rajashree Chinchole, 2015 10th International Conference on Computer Science & Education (ICCSE)
- [3] Biometric Face Recognition Payment System , Surekha. R. Gondkar Saurab. Dr. C. S. Mala International Journal of Engineering Research & Technology NCESC- 2018 Conference Proceedings
- [4] Facial Recognition in Banking – Current Applications, Niccolo Mejia,2019 Conference Proceedings
- [5] "Face Detection and Recognition for Bank Transaction ", International Journal of Emerging Technologies and Innovative Research, Sudarshan Dumbre, Shamita Kulkarni, Devashree Deshpande, P.V.Mulmule Journal of Emerging Technologies and Innovative Research 2018
- [6] Continuous User Identity Verification Using Biometric Traits for Secure Internet Services, Dr.SHAIK ADBUL MUZZER, 2GOSALA SUBHASIN
- [7] Skin color-based Face detection Method, Devendra Singh Raghuvanshi,Dheeraj Agrawal
- [8] Face Detection system based on retinal connected neural network (RCNN), Rowley, Baluja and Kanade
- [9] Combining Skin Color based Classifiers and HAAR Feature using VJ Algorithm, N.Gobinathan, Abinaya and Geetha. P
- [10] Face Detection and Recognition for Bank Transaction, Sudarshan Dumbre<sup>1</sup>, Shamita Kulkarni<sup>2</sup> , Devashree Deshpande<sup>3</sup> , Prof P.V.Mulmule<sup>4</sup>
- [11] ‘Haxby, J.V., Ungerleider, L.G., Horwitz, B., Maisog, J.M., Rapoport, S.I., and Grady, C.L. (1996). Face encoding and recognition in the human brain. Proc. Nat.Acad. Sci. 93: 922 – 927
- [12] Li, S. Z., & Jain, A. K. (Eds.). (2011). Handbook of Face Recognition. Springer.
- [13] Jain, A. K., Ross, A., & Nandakumar, K. (2011). Introduction to Biometrics. Springer.

- [14] Zhao, W., Chellappa, R., Phillips, P. J., & Rosenfeld, A. (2003). Face recognition: A literature survey. *ACM Computing Surveys*, 35(4), 399–458.
- [15] Yang, J., Lei, Z., & Li, S. Z. (2014). Learn face representation from scratch. *arXiv preprint arXiv:1411.7923*.
- [16] Sun, Y., Wang, X., & Tang, X. (2015). Deeply learned face representations are sparse, selective, and robust. *CVPR*.
- [17] Zeng, Y., Luo, X., & Xu, F. (2020). A study on the application of facial recognition in mobile payment systems. *IEEE Access*, 8, 147278–147288.
- [18] Kshetri, N. (2020). Privacy and security issues in facial recognition-based payment systems. *IT Professional*, 22(3), 56–62.
- [19] Ali, A., & Khan, S. Z. (2019). FacePay: A facial recognition-based payment system. *Proceedings of the 2019 International Conference on Biometric Engineering and Applications*.
- [20] Guo, Y., Zhang, L., Hu, Y., He, X., & Gao, J. (2016). MS-Celeb-1M: A dataset and benchmark for large-scale face recognition. *ECCV*.
- [21] Wang, M., Deng, W., Hu, J., Peng, J., & Tao, X. (2021). Deep face recognition: A survey. *Neurocomputing*, 429, 215–244.
- [22] Alibaba Cloud. (2020). Facial Recognition Technology in Alipay: Secure and Seamless Payments.
- [23] McMorrow, R. (2018). China’s facial-recognition payment revolution. *Financial Times*.
- [24] Wang, X., Liu, Y., & Wu, J. (2022). Security enhancement for facial recognition payment systems using multi-factor authentication. *Journal of Information Security and Applications*, 66, 103141.
- [25] European Commission. (2021). Facial Recognition and Biometrics in Financial Services: Ethical and Regulatory Considerations.

## A2.2 Bibliography

- [1] [https://youtu.be/z\\_dbnYHAQYg?feature=shared](https://youtu.be/z_dbnYHAQYg?feature=shared)
- [2] <https://medium.com/analytics-vidhya/haar-cascades-explained-38210e57970d>
- [3] <https://flask.palletsprojects.com/>
- [4] <https://www.sqlite.org/>
- [5] [https://docs.opencv.org/4.x/dc/dc3/tutorial\\_py\\_face\\_recognition.html](https://docs.opencv.org/4.x/dc/dc3/tutorial_py_face_recognition.html)
- [6] <https://developer.mozilla.org/>
- [7] <https://www.python.org/doc/>