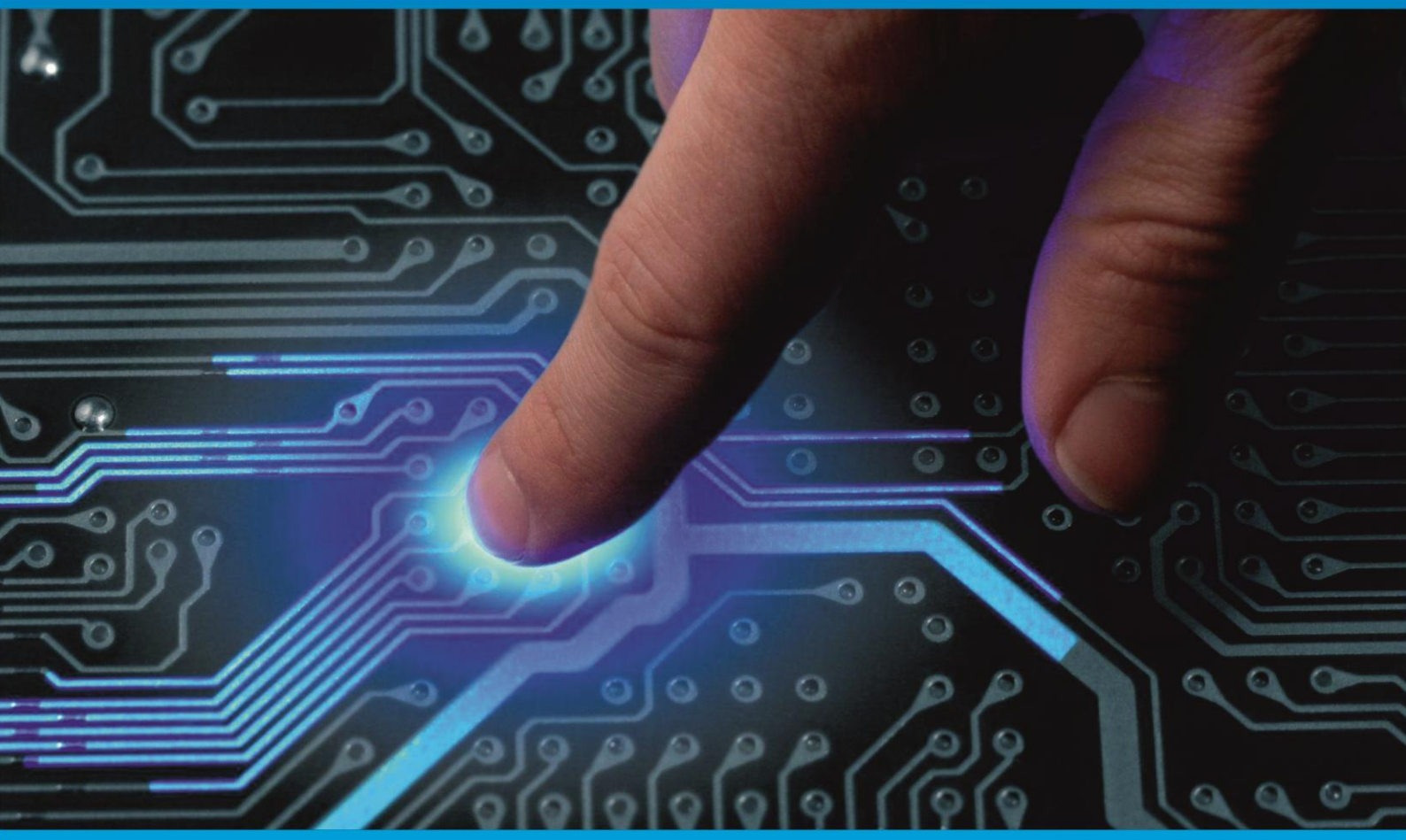
IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 8, Issue 10, October 2020

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 7.488

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 10, October 2020 ||

Development of Automated Teller Machine Authentication System (ATMAS) Software

Md Razeenuddin Mehdi[1], Uttarayan Mondal[2], Md Adil Reza[3,]

Department of Computer Science, Ram Krishna Mahato Government Engineering College, Purulia, India[1,2,3]

ABSTRACT: Automated Teller Machine(ATM) has become a vital part of our daily transactions since customers can access their bank deposit or card details to perform various financial transactions, most specifically cash withdrawals and balance checking through the medium of ATM. Additionally, ATMs are important to travelers as one can withdraw cash in foreign countries, thanks to ATM. ATM skimming and fraud are major security breaches are frequent for quite some time. As currently, the PIN on the card is the only security layer guarding customer's cash deposits. The work aims to enhance the standard of security concerning ATMs through means of creating a multibiometric authentication system comprising of a 6 digit one-time password(OTP) being sent to the registered mobile number and finally performing facial recognition along with blink detection is used to verify the identity of the customer. Furthermore, this system will free the customer from the burden of remembering PIN as the random OTP generated itself acts as the PIN. If an unauthorized user is detected, an alert SMS will be sent to the PIN admin.

KEYWORDS: ATM, Facial Recognition, Local Binary Pattern Histogram, OpenCV, Trainer, Detector.

I. INTRODUCTION

The level of security is a major concern when it comes to ATM (Automated Teller Machine). In recent times we have seen vast technological advances across the globe. Undoubtedly, these technological innovations have brought about plenty of facilities, at the same time, they have also paved the way for plenty of cybercriminals. Due to the lack of security measures in ATMs, they become the target of robbers. Some of the attacks on ATM System are described as follows:

- Eavesdropping: The ATM static four-digit PIN can easily be spied upon and thus can be accessed by obtaining the ATM card leading to serious consequences.
- Spoofing: There is a possibility that, when a user enters the PIN during the transaction process, a hacker might fake as the authorized site and prompts the user to re-enter PIN due to a system error. When the user complies with this instruction the hacker stores the data and uses it for his future ill intentions.
- Skimming: A skimmer is a card reader that can be disguised to look like part of an ATM. Now, this attachment can be used to collect card numbers and PIN codes, which are then replicated into counterfeit cards.
- Brute-force attack: The current static four-digit pin can be cracked using a brute force approach, as it takes feasible 9999 attempts to succeed. In Automated Teller Machine Authentication System (ATMAS), a randomly generated alphanumeric OTP is sent to the registered number to be used as the PIN code, thus increasing the security and reducing the chances of cracking the code using brute force. [1]

Biometric authentication can be a viable solution to the problem concerning ATM security. In the Automated Teller Machine Authentication System(ATMAS) after the customer swipes the card, a six-digit alphanumeric OTP will be sent to the customer's registered mobile number which itself acts as the PIN. Finally, a multibiometric authentication system consisting of facial and recognition and blink detection is implemented to verify the identity of the user. In case of a failed login attempt, a warning message will be sent to the original cardholder. ATMAS also cannot detect two faces at the same time thus only one person can access the ATM at a given time.

II. BACKGROUND

The core modules of the Automated Teller Machine Authentication System (ATMAS) are the OTP and facial recognition modules. The outgoing OTP message is sent by making an HTTP POST to Twilio's Message resource via the REST API. When Twilio receives our request to send the OTP message via the REST API, it will check whether we have included a valid Twilio phone number to act as the sender. The receiver of the SMS should also be a Twilio verified caller ID. Twilio will then either queue the SMS or return the HTTP error in its response to your request

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 10, October 2020 ||

(further described under the OTP system implementation - section IV-A). [2]

The face recognition system is based on the concept of texture analysis of the image. There exist algorithms like Principal Component Analysis or PCA, Fisher face or Local Discriminant Analysis, and Local Binary Pattern Histogram or LBPH for texture analysis. The face recognition system is developed by implementing the LBPH algorithm as it is known for its computational simplicity and high discriminative potential. The development of the LBPH algorithm is the work of T. Ojala in 1996 [3]. Local Binary Patterns, or LBPs in short, basically are texture descriptors which are responsible for computing a local representation of texture. To construct the local representation each pixel, excluding the non-boundary pixels is compared with its surrounding neighborhood of pixels and calculates a label or LBP value for that pixel. Furthermore, these obtained LBP values for all the corresponding non-boundary pixels are then plotted in the form of **histogram**, known as LBP histogram. The Automated Teller Machine (ATMAS) facial recognition system is inspired from Facial Recognition and Analysis System (FRAS) developed by the Department of Computer Science at St Xavier's College (Autonomous), Kolkata.[4]

The eye blink detection is used as an aliveness detector to verify whether the user trying to access the ATM is an actual person or not. The change in the aspect ratio of the eyes is used as the underlying concept of the blink detector (further explained under the Eye Blink Detector implementationsection IV-D). [5]

III. SOFTWARE DESIGN

Automated Teller Machine Authentication System (ATMAS) software is developed as a web-based application. All the dependent configuration files, scripts, databases, and other tools are kept in one main server. The flow chart shown in Fig 2 on the following page gives a brief description of how the software works. The ATM card insertion mechanism is replicated by asking the user to enter his card number in the login field which is later assigned as his unique user id. OTP is then sent to the user's registered mobile number and finally face recognition and blink detection systems are used to verify the person and authorize his access to the account.

The proposed model of ATM is dependent on four phases, i.e ATM card, user mobile phone where the OTP will be sent, face recognition, and blink detector. These four phases play a vital role in preventing a breach of security and theft of ATMs as explained below:

- If a thief forges a counterfeit card to access the user account, an OTP will be sent to the user registered mobile number, thus, notifying him that some other person is trying to gain unauthorized access to the account.
- If somehow the thief has stolen the user's mobile phone and also created a counterfeit card, then after entering the OTP, the thief's face will not be recognized as the user's face. It is also possible that the user deactivates his phone number preventing the OTP to reach the stolen phone, thus denying unauthorized access to the account.
- On a rare occasion, where the thief has prepared a counterfeit card, stole the user's phone, and has access to images of the user's face then the blink detector will classify those images as fake since no natural blinking is detected from still images. Again security breach has been prevented.

The only possible scenario where the thief can successfully gain unauthorized access to the account is when the thief has a counterfeit card or user's ATM card, user's phone, and a face mask or a closeup video of the user blinking to bypass the face recognition system and blink detector. The situation is only possible due to the negligence of the user of not reporting a stolen or misplaced phone or ATM card and not deactivating the account on time.

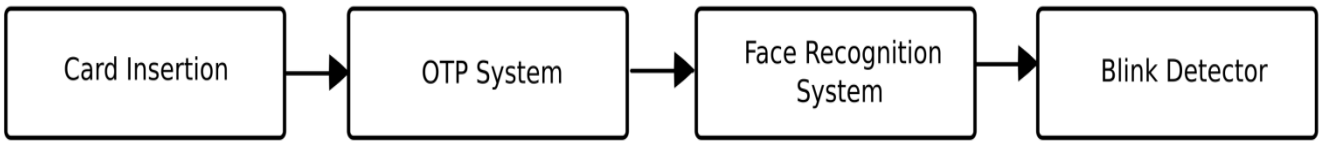


Fig 1: Phases of the model.

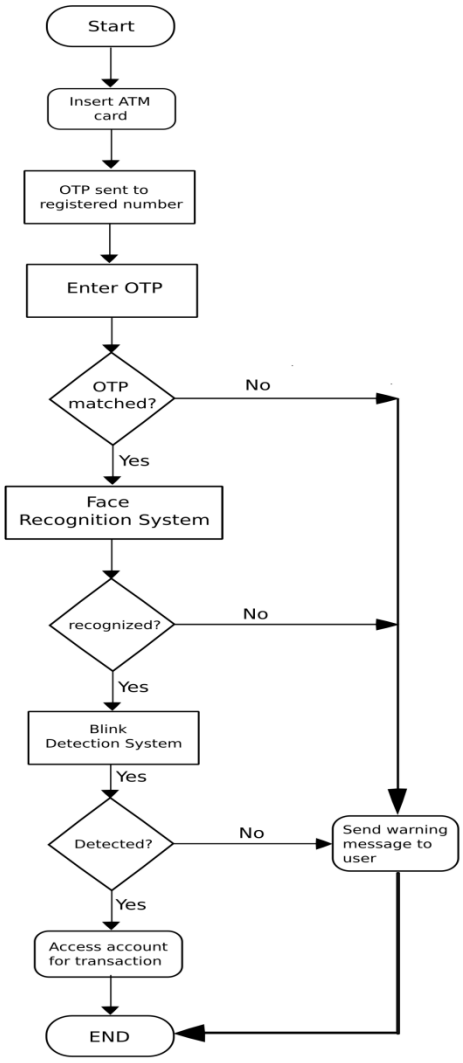


Fig 2: Flowchart of the ATMAS software

A. *Primary Modules*

There are two major modules to the Automated Teller Machine Authentication System (ATMAS) software. The various stages that the system encounters after inserting the ATM card are as follow:

1) Sending *OTP* to the user's registered mobile number using Twilio API.

2) *Facial Recognition System* is the heart and backbone of the ATMAS software which is responsible for accurately recognizing and verifying a person's face. It further comprises the following stages:

- *Data Set* consisting of a new user's face images.
- The system is then *trained* to recognize the user's face with the help of the *data set* and successfully able to differentiate the user's face from other images.
- *Detection of* the face for recognition.
- Implementing *Blink detection* to verify that an actual person's face is being detected and prevent spoofing.

B. *Conceptual Model*

The conceptual model of Automated Teller Machine Authentication System (ATMAS).

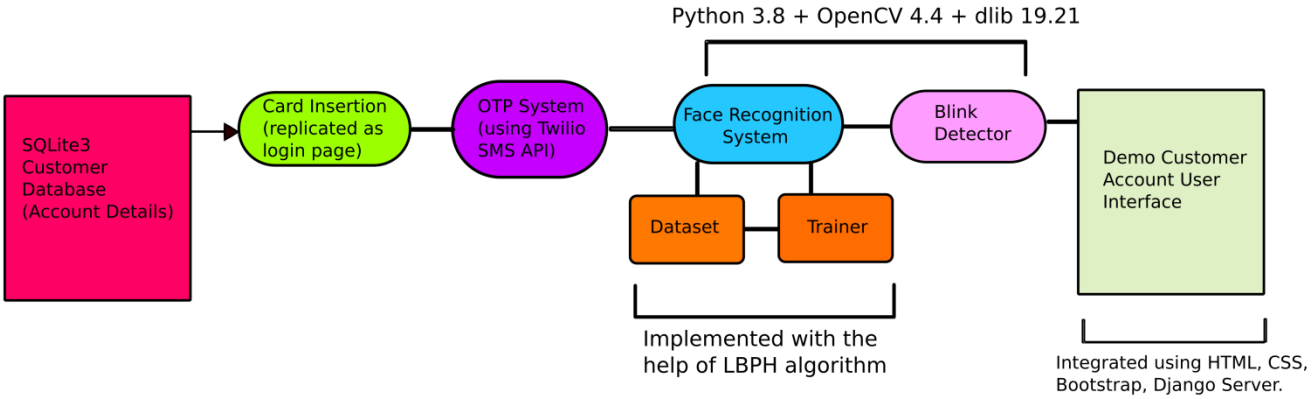


Fig 3: Conceptual Model of ATMAS software.

C. *User Interface*

The proposed ATM model has been replicated in the form of a web-based user interface, as mentioned before, the user enters the card number in the login field instead of inserting the card. All the necessary scripts, configuration files, database, and other tools are stored inside a central Django server. To access the software only three requirements are there:

- Internet Connection
- A supported browser (software was tested on Google Chrome)
- a camera (web camera or your phone camera can be used with DroidCam)

The following figure (Fig 4.) is a brief sketch of the client-server design of the system. [4]

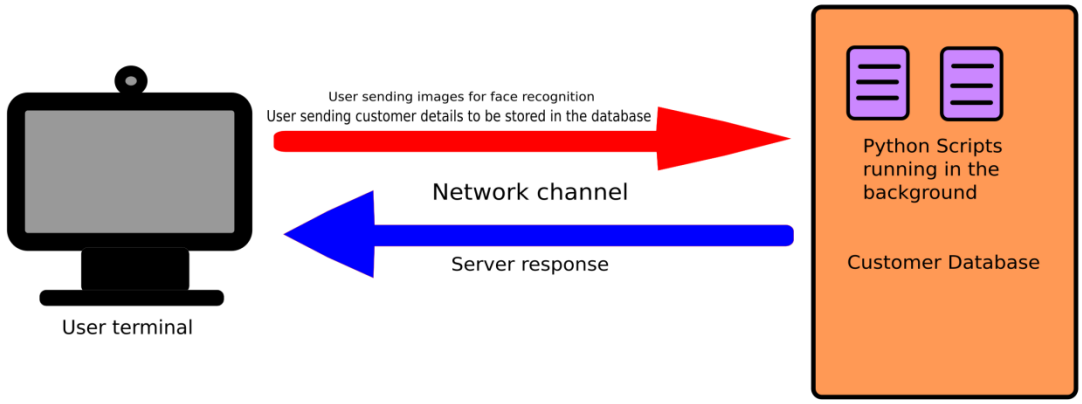


Fig 4: Client-server design of the system.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 10, October 2020 ||

IV. SOFTWARE IMPLEMENTATION

A. OTP System Implementation

The OTP system requires a Twilio phone number with SMS capabilities. For this software, a Twilio trial number is used as the sender and the receiving numbers (phone numbers of the customers) are added as verified caller IDs through the Twilio dashboard. The pseudocode for generating a random six-digit alphanumeric OTP is as follows:

i Initializing a string variable with alphanumeric characters.
ii Initialize two variables, one for Twilio Auth SID and other for Auth token.
iii Calling the Client method from Twilio REST API.
iv Generating a 6-digit string with alphanumeric characters and concatenating it with the otp string variable.
v Implementing the Twilio REST API to send the OTP to the verified caller ID.

The above algorithm has been implemented using Twilio's REST API. An outgoing OTP SMS is sent by making an HTTP POST request to Twilio's message resource. An instance of the Message resource is created by specifying the Body, To, and From parameters of the user's message. The API is adopted from the official Twilio's documentation and can be found in [2]. The following figure (Fig 5) shows the result of the execution of the otp.py script.

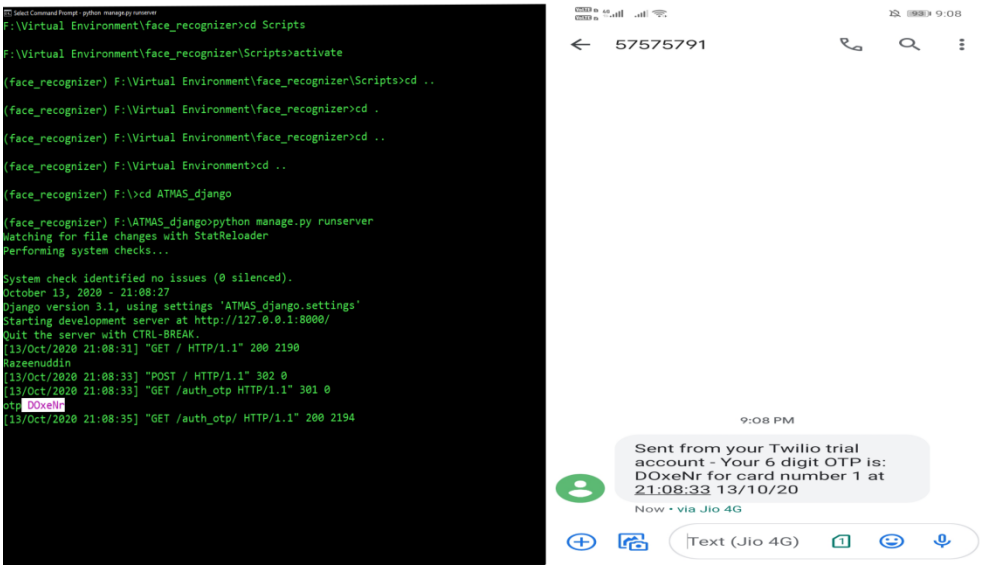


Fig 5: Output after execution of the otp.py script.

B. Implementation of LBPH algorithm

The LBPH algorithm is implemented using the Binarypattern function which is responsible for getting the Local binary pattern of an RGB image. Initially, an RGB image is converted into a grayscale image and the function Binarypattern is called using the grayscale converted RGB image. The algorithm plots the LBP histogram as the output. The LBPH algorithm has been implemented with the help of Python v3.8 [8], Matplotlib v3.3 [9], Intel OpenCV v4.4 [10], and NumPy v1.19 [11]. The program structure of the above LBPH algorithm has been adopted from an open-source public Github repository [5]. The LBPH algorithm is explained below:

i Initially, the local binary pattern array is initialized with 0.
ii A kernel of size 3*3 is taken because the input grayscale image will have a center pixel at the location (1,1) and the rest of the 8 pixels surrounding it will be considered as neighboring pixels.
iii A nested for loop is executed, with the outer loop to iterate over the height of the image, and the inner loop to iterate over the width of the image.
iv The entire grayscale image is read in 3*3 format with the center pixel assigned at the index (1,1).
v Comparing the neighboring pixels with the center pixel. If the neighboring pixel value is greater than the center pixel, 1 is assigned to that pixel, else 0 is assigned.
vi Converting the values of the newly assigned neighboring pixels to their corresponding decimal values.
vii The steps from (iv) to (vi) are repeated for all of the non-boundary pixels of the image.

viii The histogram is plotted using the vector form of image pixels.
ix The subplot consists of the original grayscale image and local binary pattern alongside the LBP histogram.

As mentioned above, the source code of the above algorithm is adopted from a public GitHub repository and can be found in [5]. The above algorithm is incorporated inside the plot.py script. To produce the LBP histogram, the *plot.py* script needs to be executed.

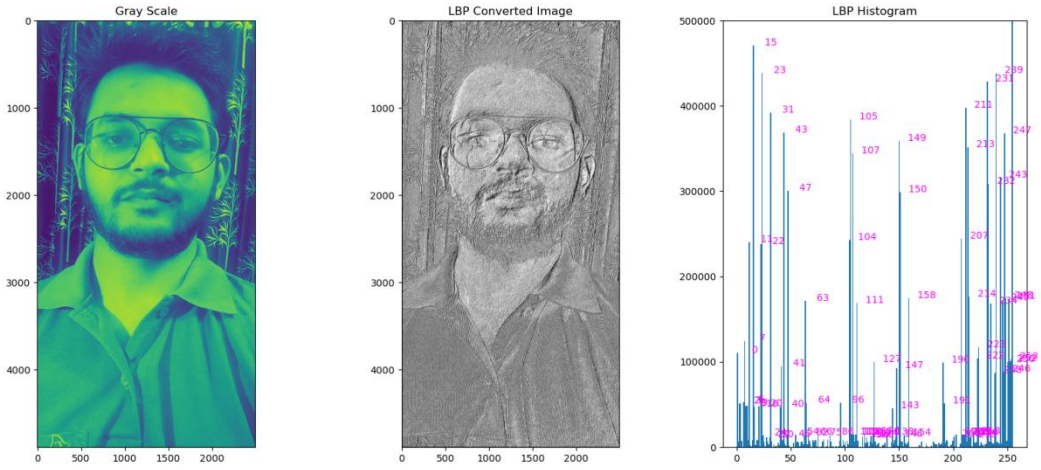


Fig 6.1: Output after execution of plot.py.

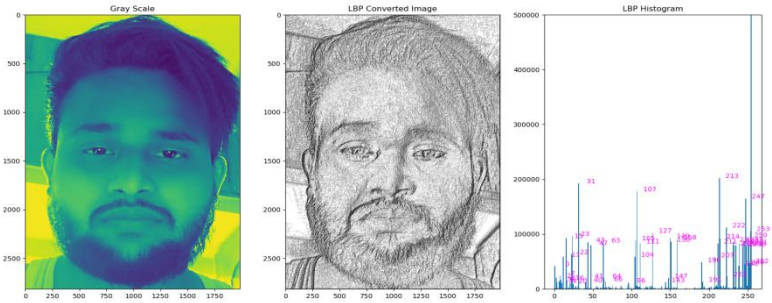


Fig 6.2: Output after execution of plot.py.

The above figures (Fig 6.1and 6.2) shows the plotted LBP histogram after executing *the plot.py* script.

C. Implementation of the Face Recognition System

The face recognition system of the ATMAS software is developed using Python v3.8 [8], Intel OpenCV v4.4 [9], dlib v19.21 [12], and Django v3.1 [13]. The database used for the face recognition system is the default Django configured SQLite database [14]. The User ID (card number) is stored as the primary key and other account details as attributes.

Following are the primary modules of the face recognition system of ATMAS software:

- *Create Data Set Module:* The module is responsible for creating a data set of the face images of a person to be detected. Haar feature-based cascade classifier is used to detect the frontal face of a person. Multiple images of the face of the user are captured at the rate of one frame per second, these images are then converted into grayscale images and then stored in the data set. LBPH algorithm is then applied in the training phase to analyze these images.

- *Trainer Module:* The trainer module is incorporated alongside the *Create Data Set* module. During the training phase of the model, the trainer module receives the analyzed data of each image from the data set after implementing the LBPH algorithm. A complete analysis is then performed to reconcile patterns in the matrices so faces can be detected for recognition and distinction. The trained data is then stored in the form of a *trainer.yml file.*
- *Face Recognizer Module:* The Face Recognizer Module is integrated alongside *Create Data Set* and *Trainer* modules. This module detects and tries to recognize the face of the user in real-time with the help of the trained data. The blink detection system is implemented if the face gets successfully recognized. If more than 200 frames of "Unknown" or "Fake" faces are detected then the user will be redirected to the initial card insertion page and a warning message will be sent to the original cardholder notifying him of the unauthorized access.
- *Integration of the Face Recognition System with Django SQLite database:* The default Django configured SQLite database is integrated with the Face Recognition System which is responsible for storing all the necessary account details. The images of the user in the data set consists of their card numbers as their unique user id in their filename. When the user's face gets recognized the ID (card number) is mapped to the database. Account details comprising of the user's name, residence, mobile, and other data are then fetched from the database using SQL statements.

D. Implementation of Blink Detection System

The Blink Detection System is integrated within the facial recognition system to prevent spoof and unauthorized access to the account. The system is implemented using Python v3.8 [8], Intel OpenCV v4.4 [9], dlib v19.21 [12]. The system applies facial landmark detection to localize and represent the eye region of the face. The following figure (Fig 7) shows the implementation of facial landmarks to detect the facial structure of a person's face using the shape prediction method.

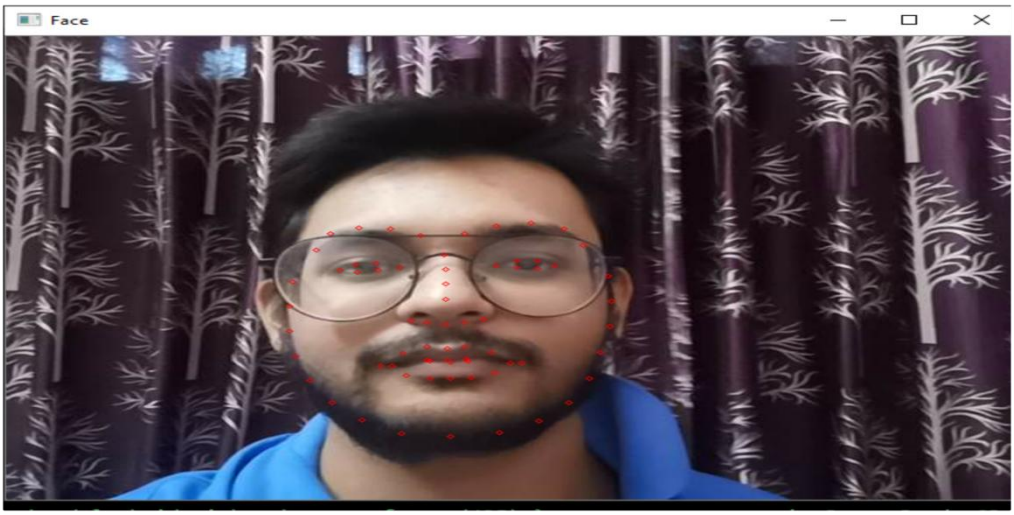


Fig 7: Facial Landmarks.

The facial landmarks are shown in red color in the above figure (Fig 7). Each eye is represented by six facial landmark coordinates. The assignment of coordinate starts at the left corner of the eye and then working around clockwise to the remainder of the region as shown in the following figures (Fig 8.1 and Fig 8.2).

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 10, October 2020 ||

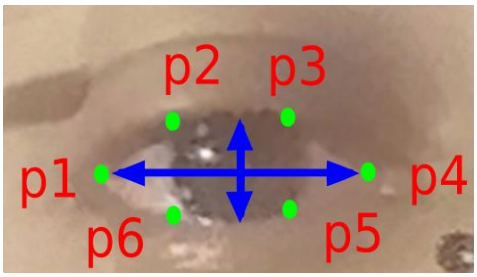


Fig 8.1: Facial landmarks while the eye is open.

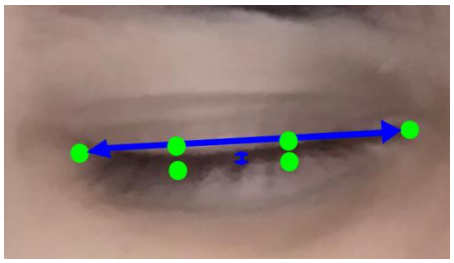


Fig 8.2: Facial landmarks while the eye is closed.

In figure 8.1 when the eye gets closed the length of the vertical line reduces, whereas when the eye is open the length of the vertical line is comparatively larger. Hence, it is evident from the above figures (Fig 8.1 and Fig 8.2) that a relationship needs to be established between the width and height of these coordinates to derive an equation for the eye blink ratio. The following equation shows the relationship between the width and height of these coordinates.

$$BlinkRatio_{lefteye} = \frac{|p1 - p4|}{midpoint(|p2 - p6| + |p3 - p5|)}$$

Similarly, the blink ratio of the right eye can also be computed. The average blinking ratio of the two eyes can be computed by halving the sum of the blink ratio of the left and right eyes. A threshold value of the average blink ratio is then adjusted to detect the natural blinking of the eyes. The Blink Detection System will prevent spoofing and unauthorized access to the account even if the face image, ATM card, and the mobile phone of the user get compromised somehow. The methodology of the Blink Detection System has been adopted from an online blog which can be found in [6].

$$AverageBlinkRatio = \frac{|BlinkRatio_{lefteye} + BlinkRatio_{righteye}|}{2}$$

E. Implementation of ATMAS software

The ATMAS software is produced after obtaining decent results in the individual unit testing of the various constituent programs. The various constituent systems of the software are primarily coded in Python v3.8 [8], Intel OpenCV [9], and the database implemented is the default Django configured SQLite [14]. The end-user interface of the software is customized and incorporated with the help of Django v3.1[13], HTML, Bootstrap, and CSS [15].

V. SOFTWARE TESTING

The different modules of the ATMAS software are integrated after undergoing individual unit testing consisting of various test cases. The software is then incorporate with the web to test the end-user interface.

The following figures (Fig. 9.1, Fig. 9.2,and Fig. 9.3) show the working of the ATMAS software.

Fig 9.1: Data Set of the model.

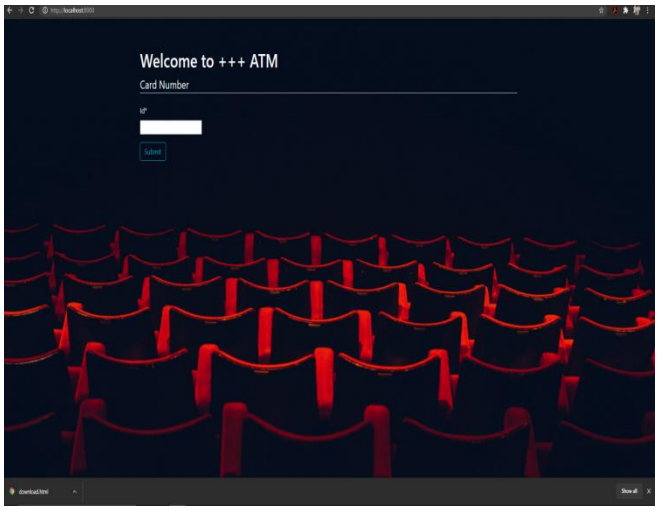


Fig 9.2: The card login page.

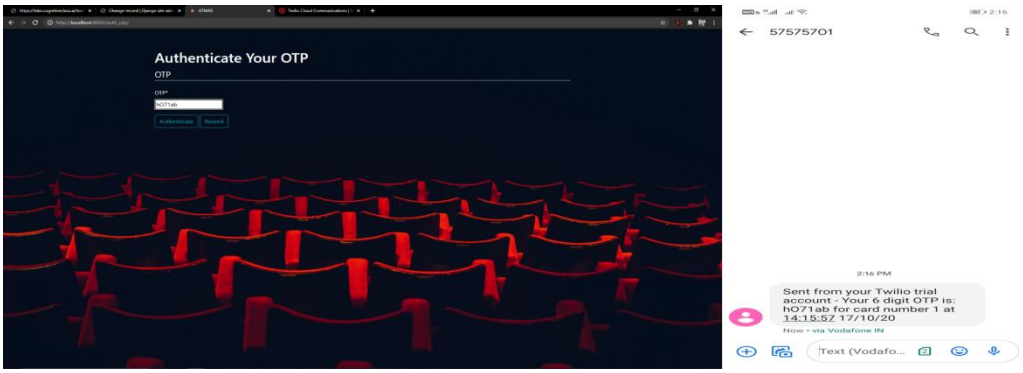


Fig 9.3: OTP system of the ATMAS software.

Figure 9.1 shows the data set formed after the execution of the *Create Data Set Module.* The data set consists of more than 40 images of the faces of two users capture at one frame per second. Figure 9.2 shows the card insertion phase of the model which is replicated as a single field login page where the user is supposed to enter his card number. The entered card number will then act as the unique user ID of the user. Figure 9.3 shows the application of the OTP system of the ATMAS software which is the second stage of the authentication process.

The following figures (Fig. 9.4, Fig. 9.5, and Fig. 9.6) shows the application of the Face Recognition System along with the Blink Detection System of the ATMAS software. In case of an unauthorized access a warning message will be sent to the user's registered mobile number. The following figure 9.4 shows a warning message being sent after detection of "Unknown" and "Fake" faces.

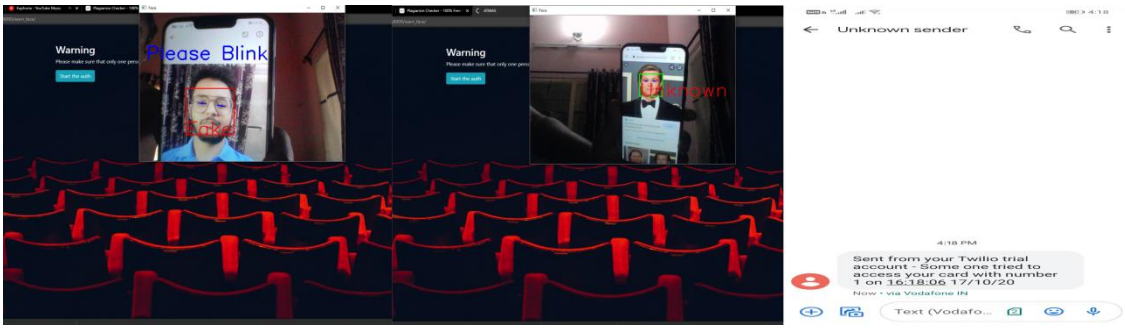


Fig 9.4: Warning message sent in case of unauthorized access.

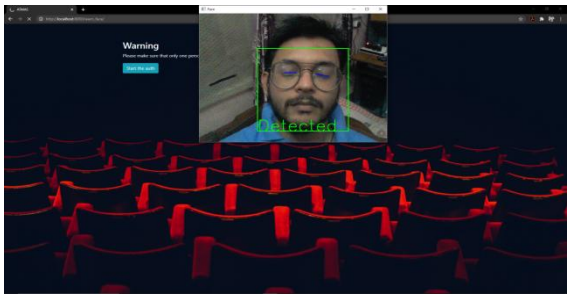


Fig 9.5 Face recognized while blinking.

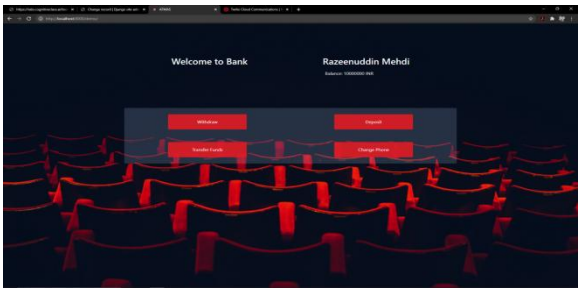


Fig 9.6 Demo user interface of the customer account.

VI. SIMULATION SETUP

Operating System: The ATMAS software is tested on Arch Linux (current release: 2020.11.01Included Kernel: 5.9.2) and Windows 10.

Webcam: The webcam information for the ATMAS software is shown below:

i Webcam resolution: 640 X 480
ii Webcam Aspect Ratio: 1.33
iii Webcam Frame Rate: 30

Browser used: The web browser used for the end-user interface of the ATMAS software is CHROME V86.0.4240.111

VII. RESULTS AND DISCUSSION

After performing unit testing of the various modules, integration testing with the end-user interface, and testing of the ATMAS software as a whole, it can be concluded that the optimal conditions for achieving maximum accuracy from the software are as follows:

- *Confidence Value:* 45 to 65
- *Light Intensity:* Very High to Medium
- *Face distance from the camera:* 0.5 ft. To 3ft.
- *Average blinking ratio:* 6.0 to 8.0

The testing of the ATMAS software is done manually, and the testing scenarios are adopted from the Face Recognition and Analysis System (FRAS) software [4]. The confidence value of the Face Recognition System of ATMAS software depends on the following factors:

- Quantity of the training images
- Light Conditions
- Face distance from the camera
- dlib frontal face detector

The average blinking ratio of the Blink Detection System depends on the following factors:

- Light Conditions
- Face distance from the camera
- Facial landmark points representing the eyes

The following Table 1 and Fig. 10.1 shows the relationship between the confidence value and the accurate recognition rate of the ATMAS software. The average blinking ratio threshold value is set to 0.0 as the testing case is purely focused on the ranges of the confidence value at which the face of the user gets successfully recognized.

Confidence Value	Test Result
0-20	Anomalous, Shows "Unkown" most of the times.
20-45	Correct Results, but sometimes shows "Unknown".
45-65	Consistently recognizes face accurately.
65-80	Needs high intensity of light to obtain correct results.
80-110	Needs very high intensity of light to obtain correct results.
110+	Anomalous.

Table 1: Relation between Confidence Value and Accuracy.

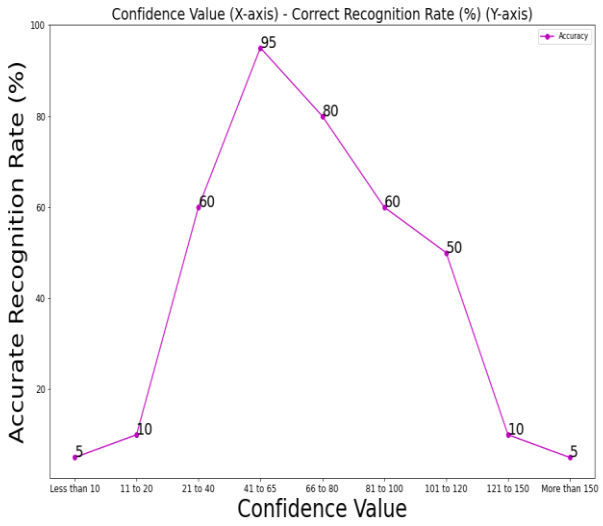


Fig 10.1: Graph depicting the relation in Table 1.

The average blinking ratio threshold value is set to 8.0 to test the accuracy of ATMAS software with the variation of intensity of light (Very high is above 5000 lumens and very low is below 100 lumens). The following Table 2 and Fig 10.2 depicts the relation between the accuracy of the ATMAS software and light intensity of the surrounding.

Light Intensity	Test Result
Very Low	Result directly dependent on face distance from camera. Shows "Fake" and "Unknown" sometimes.
Low	Result directly dependent on face distance from camera. Accurate results most of the time.
Medium	Result directly dependent on face distance from camera. Accurate Results
High	Consistently accurate results.
Very High	Consistently accurate results.

Table 2: Relation between Light Intensity and Accuracy of software.

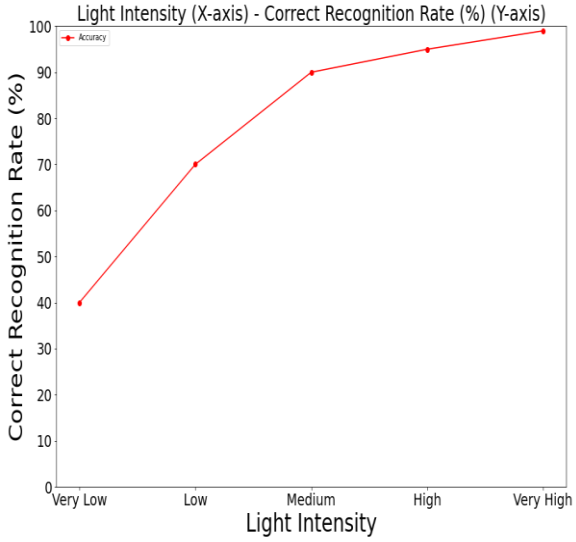


Fig 10.2 Graph depicting the relation in Table 2.

The face distance from the camera is also an important factor while accuracy recognizing the face and detecting the eye blink. In the case where the distance between the face and camera is above 3 feet, there is a chance that the vertical width of the facial landmark coordinates of the eyes becomes zero. When the vertical width becomes zero the average blinking ratio is set at a value of 100.0, thus any detected face at a distance more than 3 feet will be considered "Fake" or "Unknown". The average blinking ration threshold value is set at 8.0. The following Table 3 and Figure 10.3 shows the relation between the accuracy of ATMAS software and the face distance from the camera.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 10, October 2020 ||

Face Distance from camera (in ft.)	Test Result
Below 0.5 feet	Shows "Unknown" or "Fake" most of the time.
0.5 feet to 1 feet	Uniformly correct results.
1 feet to 3 feet	Uniformly correct results.
Above 3 feet	Anomalous.

Table 3: Relation between the Face distance from camera and the Accuracy

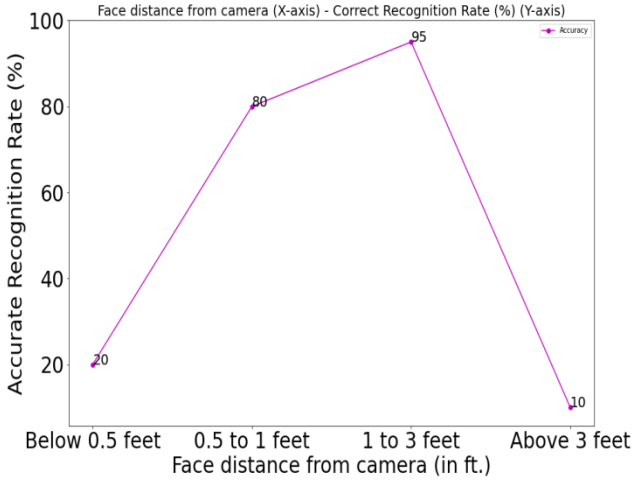


Fig 10.3: Graph depicting the relation in Table 3.

The following figures (Fig. 10.4. and Fig. 10.5) shows the graphical representation of the average blinking ratio while eyes are open and closed respectively.

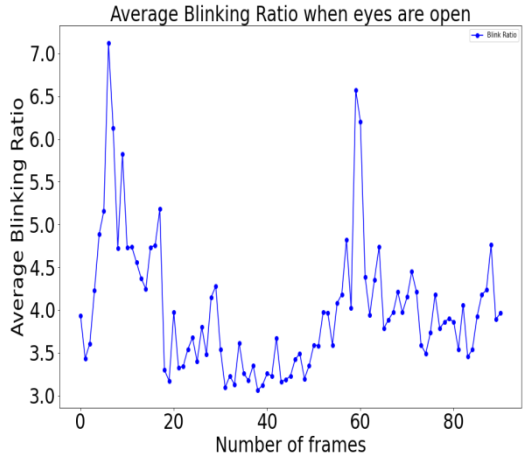


Fig 10.4: Average Blinking Ratio when eyes are open.

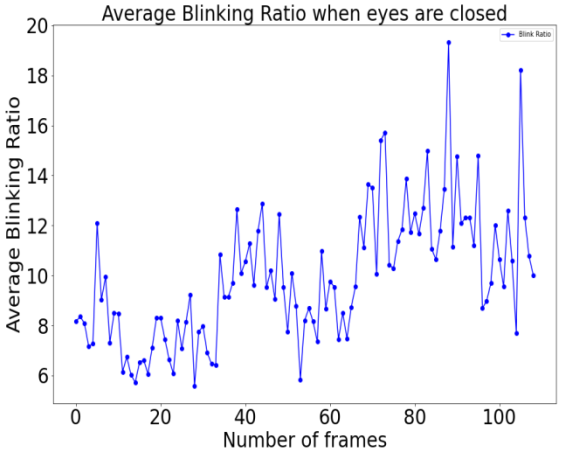


Fig 10.5: Average Blinking Ratio when eyes are closed.

VIII. CONCLUSION

The development of the ATMAS software was undertaken with the sole objective of enhancing the security level of the ATMs and with the above results obtained, it can be concluded that the ATMAS software has achieved its purpose. The software is still nowhere near perfect but is still an improvement over the current ATM model. The following are some flaws of the proposed ATM model.

- The major drawback of the system is when a particular network service is down, then there is no way for the user to receive the OTP. The only alternative to this situation is for the user to change his registered mobile number.
- The proposed model of ATM is heavily dependent on the intensity of light and camera quality. In a situation where the camera gets destroyed, no authorization will be possible until the camera is repaired.

The ATMAS software was originally developed to enhance the security level of modern-day ATM s, but the software can be further improved and applied in other fields in the future. The source code of the ATMAS software can be found in [7]. The following are the future scope and fields in which the ATMAS software can be applied.

- Enhancing household security through means of face recognition and blink detection
- Implementing the ATMAS software for online banking authentication
- Incorporating an emotion recognition feature to detect malicious intent to reduce the crime rate.

IX. ACKNOWLEDGMENT

We would like to extend our gratitude to the Department of Computer Science at Ram Krishna Mahato Government Engineering College, Purulia for helping with the development of this project as well as to our parents for supporting us during this endeavor. We would also like to thank the various online resources that we referred to during the development of this software.

REFERENCES

1 Moshin Karovaliya, Saifali Karedia, Sharad Oza, Dr.D.R.Kalbande, “Enhanced security for ATM machine with OTP and Facial recognition features”, Computer Engineering Department, Sardar Patel Institute of Technology, International Conference on Advanced Computing Technologies and Applications (ICACTA-2015), 2015.

2 Twilio official documentation: https://www.twilio.com/docs/sms/tutorials/how-to-send-sms-messages-python, 2010.

3 Timo Ojala, Matti Pietkainen, David Harwood, “A comparative study of texture measures with classification based on featured distributions, Pattern Recognition”, Volume 29 Issue 1, January 1996.

4 Ashin Guha Majumdar, Anurag Roy, Suvasish Dasgupta, Shalabh Agarwal, “Developing a Face Recognition and Analysis System (FRAS) and its Applications”, Computer Engineering Department, St Xavier’s College (Autonomous), Kolkata, Volume 5 Issue 5, May 2017.

5 Public github repository, Author: Saloni Bhatia Dutta; https://github.com/salonibhatiadutta/To-get-the-local-binary-pattern-LBP-of-image-and-draw-its-histogram/blob/master/gray_image_conversion.ipynb, October, 2019.

6 Pysource computer vision blog, Author: Sergio Canu, Title: “Eye Blinking detection – Gaze controlled keyboard with Python and Opencv p.2”; https://pysource.com/2019/01/10/eye-blinking-detection-gaze-controlled-keyboard-with-python-and-opencv-p-2/, January, 2019.

7 ATMAS, Authors: Md Razeenuddin Mehdi, Uttarayan Mondal, Md Adil Reza, Title: "Automated Teller Machine Authentication System (ATMAS)"; https://github.com/razeen-cyber/ATMAS_django ATMAS source code, 2020.

8 Author: Python Software Foundatio; Title: "Python Language Reference"; https://docs.python.org/3.8/reference/ Python 3.8.2, Oct 16, 2020.

9 OpenCV Documentation, Intel, https://docs.opencv.org/4.4.0/ OpenCV 4.4.0, July 18, 2020.

10 Matplotlib Documentation, Original Author: John D. Hunter; https://matplotlib.org/3.3.1/contents.html Matplotlib 3.3.1, August 14, 2020.

11 NumPy Documentation, Original Author: Travis Oliphant; https://numpy.org/doc/stable/ NumPy 1.19.1, June 29, 2020.

12 dlib Documentation, Original Author: Davis E. King; http://dlib.net/ dlib 19.21.0, August 08, 2020.

13 Django Documentation, Django Software Foundation, https://docs.djangoproject.com/en/dev/releases/3.1/ Django 3.1, August 04, 2020.

14 Django Databases Documentation, Django Software Foundation, https://docs.djangoproject.com/en/3.1/ref/databases/#sqlite-note , August 04, 2020.

15 Bootstrap Documentation, Original Authors: Mark Otto, Jacob Thornton, https://getbootstrap.com/docs/4.1/getting-started/introduction/.

INNO SPACE

SJIF Scientific Journal Impact Factor

Impact Factor:
7.488

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details