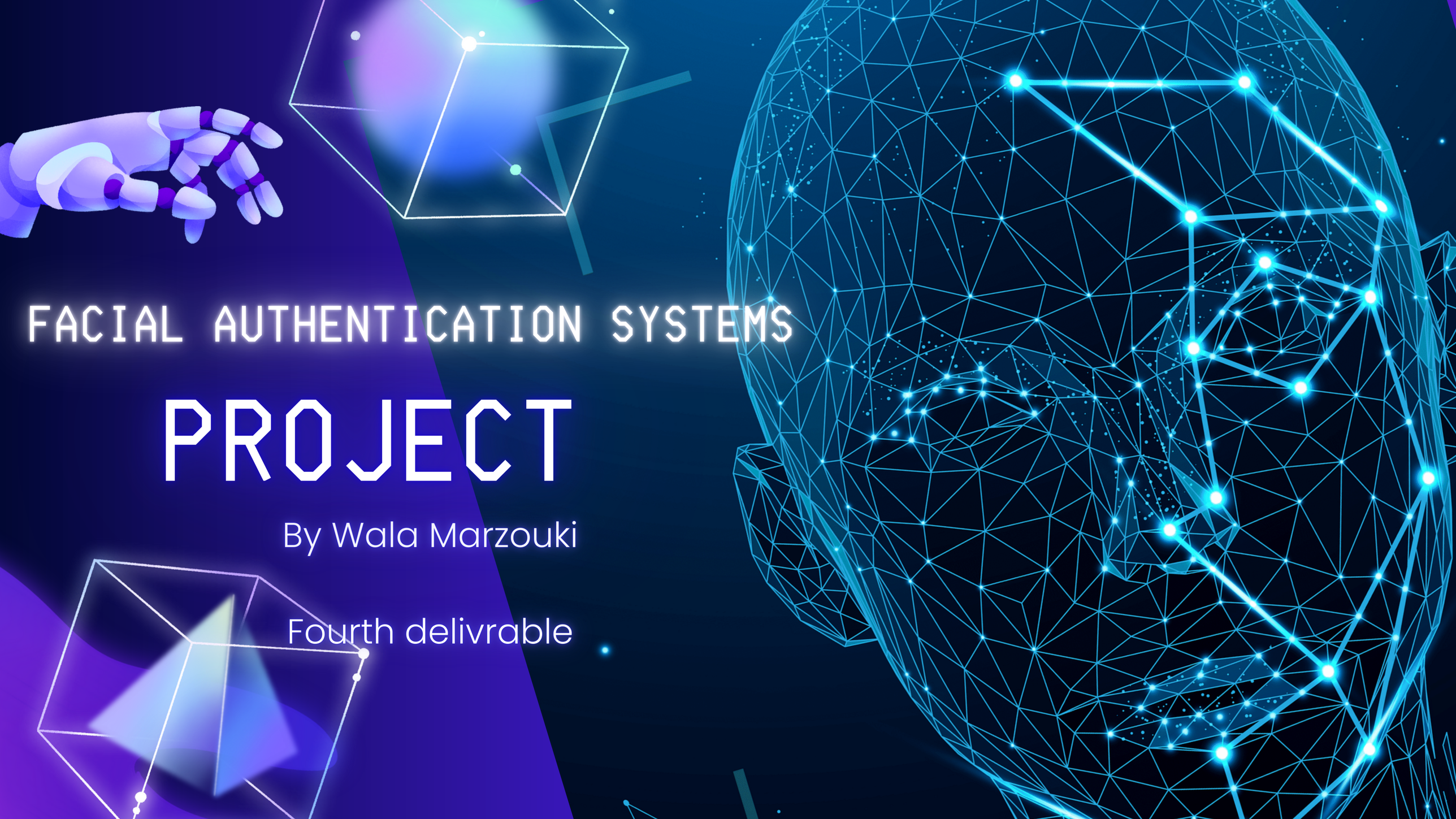


FACIAL AUTHENTICATION SYSTEMS

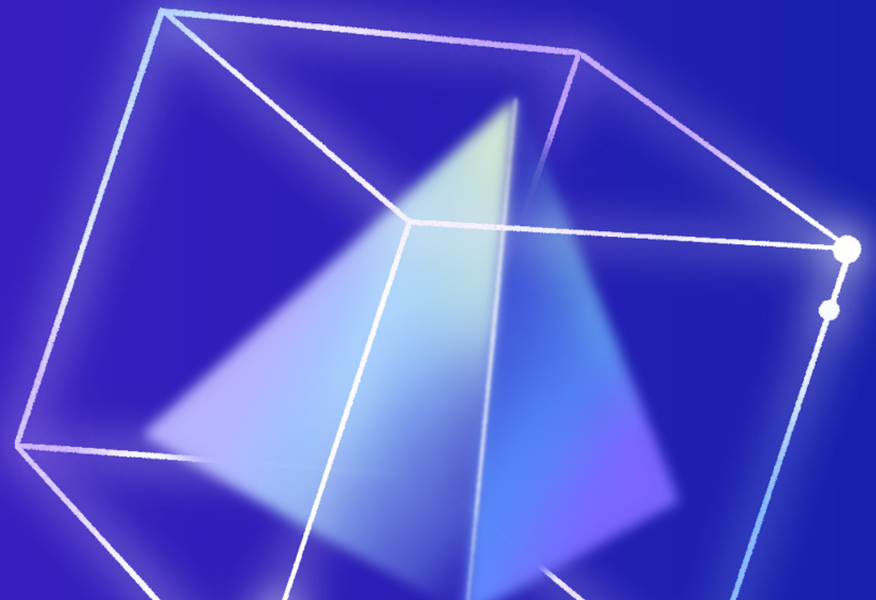
PROJECT

By Wala Marzouki

Fourth delivrable

MY FACIAL AUTHENTICATION WEB APP

TABLE OF CONTENTS



FACIAL AUTHENTICATION SYSTEM: PURPOSE AND SCOPE

The primary purpose of my Facial Authentication System is to enhance security measures by implementing a reliable and efficient facial recognition login mechanism. This system is designed to verify the identities of individuals trying to access sensitive data and restricted meetings of the corporate environment, ensuring that only authorized personnel can gain access.

This project encompasses the development and deployment of a facial authentication system using computer vision and machine learning technologies. The system captures real-time video streams to detect and recognize human faces. Once a face is detected, the system compares it against a pre-stored database of authorized users to authenticate identity. If the authentication is successful, the user gains access to the company's website, and their name and timestamp of entry are automatically recorded in an Excel sheet for attendance and security tracking purposes

TOOLS AND TECHNOLOGIES USED

In the development of the Facial Authentication System, a variety of tools and technologies were employed. Below is a detailed overview of the key tools and technologies utilized:

Hardwares

Web Cameras: web cameras are critical for their ability to capture high-resolution video which is essential for real-time video processing and facial recognition tasks.

Softwares and libraries

Python: Serving as the core programming language for the project. It offers an extensive support through its libraries, facilitating the integration of different components of the facial recognition system. It supports backend development with the libraries OpenCV for image processing and Flask for web application frameworks.

- **OpenCV (Open Source Computer Vision Library**): is a huge open-source library for computer vision (means the extraction of information from images, text, videos, etc), machine learning, and image processing. It contains over 2500 algorithms. it is instrumental in handling real-time computer vision tasks. It processes video stream data, detects faces, and extracts facial features that are crucial for the authentication process.
- **Flask**: a lightweight web framework and a Python module that lets you develop web applications easily. In this project, Flask plays a pivotal role in both constructing the web interface and managing the server-side logic. It is specifically utilized to efficiently handle HTTP requests, such as user login, video stream requests, and website redirection. Flask's session management capabilities are crucial for maintaining user states across multiple requests, allowing the system to persist user identity once logged add to that Flask’s ability to manage user sessions securely each session is encrypted using a dynamically generated secret key, safeguarding user data and authentication details effectively. Furthermore, Flask facilitates a smooth interaction between the backend, where the facial recognition processing occurs, and the frontend, where users interact with the system.
- **OpenPyXL**: is a Python library to read and write Excel files. It is utilized for managing an Excel-based logging system. It logs names and timestamps of authenticated users, thereby providing a reliable method for attendance and access control.

Machine Learning and Image Processing Tools

Haar Cascade Classifiers: It is a machine learning-based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images. Employed in this project for identifying face structures from the video frames, setting the foundation for further facial recognition processes.

Web Development Technologies:

- **HTML and CSS:** These technologies are employed to craft and style the login and main interface pages of the web face application and the website. They ensure that the user interface is not only functional but also user-friendly and accessible.
- **JavaScript:** To enhance interactivity on the client side of the web application, JavaScript is used to manage dynamic elements.

DEVELOPMENT PHASES

The development of the Facial Authentication System was strategically segmented into three key phases to ensure a systematic and efficient rollout.

Phase One

Login Interface

This phase consist of the creation of a secure login page using Flask. The development began with setting up a basic Flask application to manage user sessions and handle HTTP requests efficiently. A login form was designed using HTML and integrated into the Flask application. Users input their credentials on this form, which are then securely transmitted and verified against a pre-defined dictionary of usernames and passwords stored in the application. This verification process checks if the entered credentials match any of the stored combinations and if successful, the session is established using Flask's session management which ensures that access is granted only to authenticated users who have successfully logged in.

Phase Two

Facial Authentication Module

This phase aims to enhance security by incorporating biometric authentication. This phase leveraged the capabilities of OpenCV to implement real-time face detection and recognition. The development involved setting up a video capture mechanism using OpenCV's VideoCapture object to access webcam data. The real-time video stream was continuously processed to detect human faces using OpenCV’s pre-trained Haar Cascade classifiers. Upon detecting a face within the video frames, the system utilized histogram comparison methods to match the detected face against a pre-stored database of grayscale images representing authorized users. This database was maintained as a list of reference images, with corresponding names mapped to each image for identification. The histograms of the detected faces were compared to those of the reference images to calculate similarity scores. A threshold for similarity was set to determine successful authentication, ensuring that only faces with high correlation scores were considered as authenticated. Successful authentication triggered the recording of the user's name and the timestamp in an Excel file for attendance and security tracking by using the OpenPyXL library and the redirection to the corporate website.

Phase Three

Corporate Website Access

This phase focuses on creating a user-friendly and secure web platform for authenticated users. This final stage involved the utilization of HTML, CSS, and JavaScript to craft the front-end interface of the corporate website. The HTML provided the structural foundation, while CSS was employed to enhance the aesthetic appeal and ensure a responsive layout across various devices. JavaScript played a crucial role in enriching the user experience by enabling dynamic content updates and interactive features without needing to reload the webpage.

The development of the Facial Authentication System progressed through carefully planned phases, each designed to build upon the previous to form a comprehensive and secure system.

CONCLUSION

This document has outlined the complete development process of My Facial Authentication System designed to enhance security measures within corporate environments. By integrating advanced technologies such as Flask, OpenCV, and OpenPyXL, the system provides a strong framework for real-time facial authentication. The step-by-step development strategy, from constructing a secure login interface to deploying sophisticated facial recognition capabilities and developing a fully functional corporate website, ensures that the system is not only secure but also user-friendly.

THANK YOU!