Web-Based Facial Authentication System

INFORMATION ASSURANCE AND SECURITY

Prepared By :

- Yosr Boussarsar
- Islem Hamzaoui
- Eya Essid
- Eya Trabelsi
- Wadii Selman

TABLE OF CONTENT

INTRODUCTION

Building a web-based facial authentication system involves several stages, including planning, development, and deployment. This system typically requires various technologies and tools to perform facial recognition and authentication tasks. Here's an outline of the tools and development phases involved.

PLANNING PHASE

- **Requirement Analysis:** Determine the objectives, target users, security requirements, and functional specifications.

- **Architecture Design:** Define the system architecture, including client-side and server-side components.

- **Technology Stack Selection:** Choose appropriate technologies for the front-end, back-end, database, and machine learning.

DEVELOPMENT PHASE

Front-End

- **CSS and HTML:** Used for designing the user interface and layout of the web application.
- **JavaScript:** Used to handle user input and communicate with the server.
- **Camera Integration:** Use the WebRTC API to access the user's camera for facial scanning**.**

Back-End

- **Server-Side Frameworks:** Python
- **Web Server:** Apache to serve the web application.
- **Authentication:** JSON Web Tokens for secure authentication and session management.
- **API Integration:** PHP for communication between front-end and back-end.

Database

- **Database System:** SQL (MySQL)

PYTHON LIBRARIES

MySQL Connector: This library allows Python to connect to a MySQL database.

NumPy: NumPy is a library for numerical computing with Python. It's used here for handling arrays and mathematical operations.

os: Provides functionalities for interacting with the operating system, such as navigating directories.

Facial Recognition Libraries:
OpenCV: Open Source Computer Vision Library is a popular library for computer vision and image processing tasks. It's used for face detection and image manipulation.
dlib: a modern toolkit containing machine learning algorithms and tools for creating complex software. It's used for facial landmark detection.

PYTHON LIBRARIES

Machine Learning Frameworks:
scikit-learn: a machine learning library for Python. It's used here for splitting the dataset into training and testing sets, as well as for training and evaluating a Support Vector Machine (SVM) classifier.

Data Preprocessing: Techniques for enhancing images, such as normalization and alignment.
PIL: Python Imaging Library adds image processing capabilities to Python interpreter. We used it for loading and manipulating images.

ALGORITHMS:

Haar Cascade Classifier: This is a machine learning-based approach where a cascade function is trained from a lot of positive and negative images to detect objects in other images. In this code, it's used for face detection

Facial Landmark Detection: This is a computer vision technique that identifies key points on a face, such as the position of the eyes, nose, and mouth.

Support Vector Machine (SVM): SVM is a supervised machine learning algorithm that can be used for classification tasks.

PCA (Principal Component Analysis): Used for reducing the dimensionality of the feature vectors.

TESTING PHASE

- **Unit Testing:** Pytest (Python), for testing individual components.
- **Integration Testing:** Test interactions between components to ensure they work as expected.
- **User Testing:** Real-world tests with users to ensure usability and security.

DEPLOYMENT PHASE

- **Continuous Integration/Continuous Deployment (CI/CD):** GitHub Actions for automated testing and deployment.
- **Security Tools:** hash functions, SQL injection prevention methods, tokenization.

MAINTENANCE PHASE

- **Updates and Upgrades**: Regular updates to ensure compatibility and security.
- **Bug Fixes**: Address reported issues promptly.
- **User Feedback**: Gather feedback to improve system usability and functionality.
- **Documentation**: Maintain comprehensive documentation for developers and users.

CONCLUSION

In summary, The web-based facial authentication system solution is designed as a client-server architecture. The client side is built using HTML, CSS, and JavaScript, while the server-side is built using Python. The system uses OpenCV for facial recognition and MySQL as the database to store user information.
The development phases and tools explained above cover the typical lifecycle of a web-based facial authentication system.