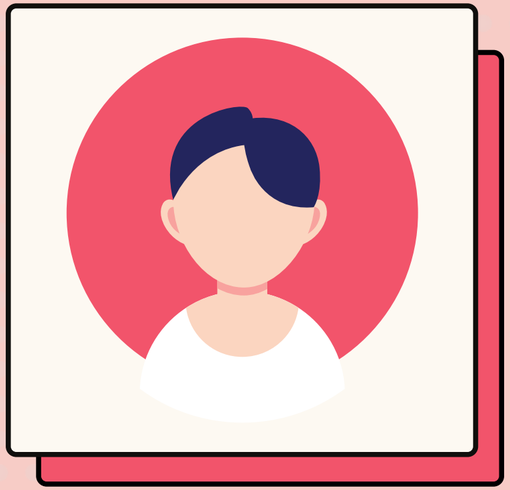
Group Project

USER AUTHENTICATION BY FACIAL RECOGNITION

PRESENTED BY GROUP 17

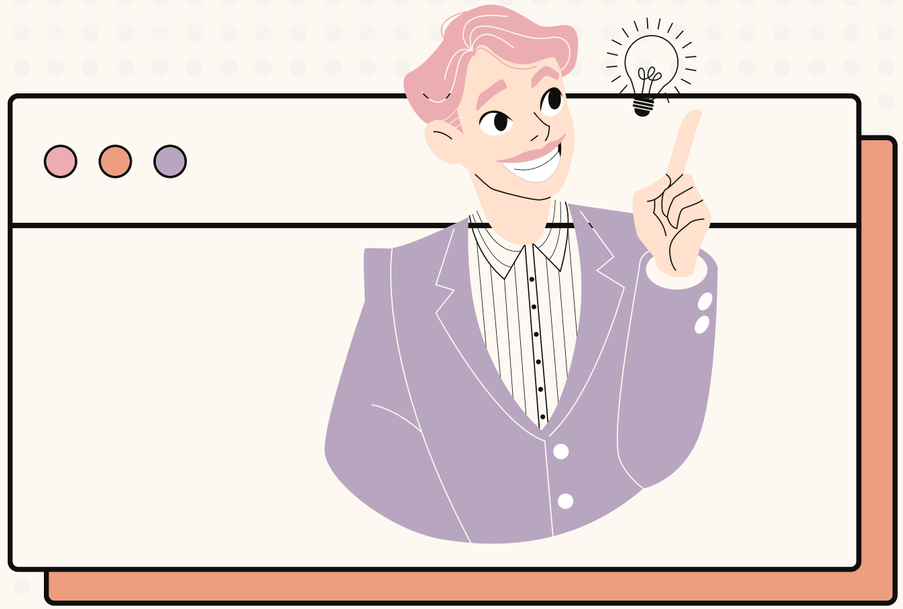
Team Members



6388113 Poomrapee Wareeboutr

6388133 Pitchaya Teerawongpairoj

6388196 Sasima Srijanya



Introduction

- The process of verifying a user's identity.
- Store the image of the user in the database.
- Comparing real-time face scanning with the image of the user's face in a database.
- If the images match, the user is authenticated and allowed to access the system.

Tech Stack

VUE.JS

Web App

PRELINE

UI Decoration

SQLAlchemy

SQLALCHEMY

Database

Flask

+ Face Recognition

FLASK

API / Server

Sign-up Workflow

INPUT INFO

Input username and face scanning

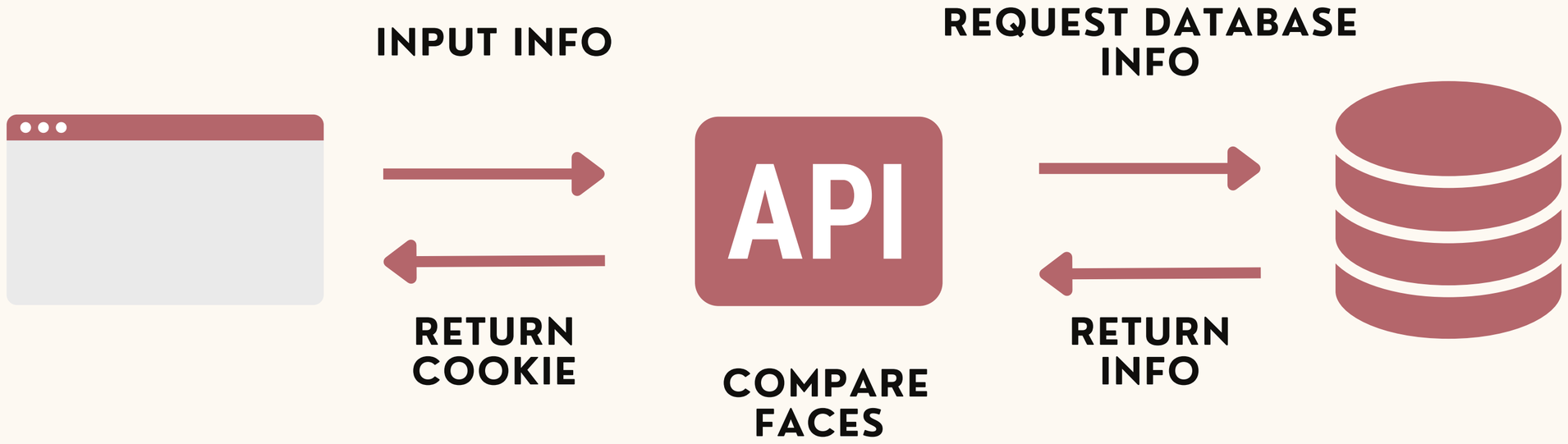
REQUEST API

Send request to save a new user info

DATABASE

Save user info in local database

Sign-in Workflow



Face compares (sign in)

```python
@cross_origin()
@app.route('/signin', methods=['POST'])
def signin():
    username = request.form['username']
    user = User.query.filter_by(username=username).first()
    if not user:
        return jsonify({ 'message': 'This username does not exist' }), 401

    input_picture = request.form['picture']
    input_picture = input_picture.split(',')[1]
    input_picture = base64.b64decode(input_picture)

    picture_of_user = face_recognition.load_image_file(user.image_path)
    user_face_encoding = face_recognition.face_encodings(picture_of_user)[0]

    input_picture = face_recognition.load_image_file(io.BytesIO(input_picture))
    input_face_encoding = face_recognition.face_encodings(input_picture)[0]
    results = face_recognition.compare_faces([user_face_encoding], input_face_encoding, 0.475)

    # See the distances between two faces
    # face_distances = face_recognition.face_distance([user_face_encoding], input_face_encoding)
    # print(face_distances)

    if results[0] == False:
        return jsonify({ 'message': 'Unauthorized' }), 401

    return jsonify({ 'message': 'Signed in successfully' }), 200
```

Save a new face (sign up)

```python
def signup():
    username = request.form['username']
    confirm_username = request.form['confirmUsername']
    if username != confirm_username:
        return jsonify({ 'message': 'The Username and confirm username does not match' }), 400

    user = User.query.filter_by(username=username).first()
    if user:
        return jsonify({ 'message': 'This username already exists' }), 401

    user = User(username=username, image_path=f'./dataset/{username}.jpg')
    db.session.add(user)
    db.session.commit()

    input_picture = request.form['picture']
    input_picture = input_picture.split(',')[1]
    input_picture = base64.b64decode(input_picture)

    # Create the ./dataset/ directory if it does not exist
    dataset_dir = os.path.join(os.getcwd(), 'dataset')
    if not os.path.exists(dataset_dir):
        os.mkdir(dataset_dir)

    # Save the image to the ./dataset/ directory
    filename = f'{username}.jpg'
    filepath = os.path.join(dataset_dir, filename)
    with open(filepath, 'wb') as f:
        f.write(input_picture)

    return jsonify({ 'message': 'Signed up successfully' }), 201
```


cout << "Enter element a" << i + 1 << j + 1 << " : ";
cin >> a[i][j];
// Storing elements of second matrix.
cout << endl << "Enter elements of
cout << endl;

Demo Part

Thank You

Do you have any questions of us?