Project Overview

This project is a web application designed to manage and track employee wellness programs, participation, and health metrics efficiently. Developed using Flask and Python for the backend, MySQL for the database, and HTML/CSS for the frontend, the system aims to streamline program management while offering valuable insights to HR and wellness coordinators.

Application Access

How to Access the Application

Launch the application by following instructions in the README.

User Credentials

Here are the credentials for testing the application, with three accounts representing different functionalities:

1. **Coordinator Account**
    - **Email**: michael.williams@gre.com
    - **Employee ID**: 103
    - **Credentials:** CRETFIT
        i. **Credentials are only needed for the Coordinator**
    - **Role**: Administrator with full privileges, including managing employees and programs.
2. **Wellness Coordinator Account**
    - **Email**: daniel.taylor@gre.com
    - **Employee ID**: 117
    - **Role**: Coordinator responsible for managing wellness programs and accessing detailed metrics.
3. **Employee Account**
    - **Email**: john.smith@gre.com
    - **Employee ID**: 101
    - **Role**: General employee with restricted access to create personal health metrics.

Key Features and Functionalities

Objectives

- **Track and Manage**: Employees' health metrics, program participation, and outcomes.
- **Streamline Access**: Enable easy data access for administrators and coordinators.

- **Secure System**: Validate users and prevent unauthorized access.

Core Functionalities

1. **Employee Management**:
    - Add, delete, and update employee information through intuitive forms.
    - Assign roles (Coordinator, Secretary, Worker) to employees.
2. **Wellness Program Management**:
    - Add, update, and delete wellness programs.
    - Store program details including start and end dates, program type, and name.
    - Assign a coordinator to each program.
3. **Health Metrics Tracking**:
    - Maintain a history of employee health records, including BMI, blood pressure, heart rate, and cholesterol measurements.
    - Provide a data table for representation and easier analysis.
4. **User-Based Navigation**:
    - Dynamic navigation bar based on user roles for a personalized experience.
5. **Error Prevention and Feedback**:
    - Validate form inputs and login credentials.
    - Use flash messages to provide feedback and prevent conflicts with the database.

Technical Implementation

Backend

- **Flask Framework**:
    - Modularized routing and templates for clean architecture.
    - Implemented RESTful principles for managing database operations.
- **MySQL Database**:
    - Designed relational tables to store employees, coordinators, programs, and health metrics.
    - Created queries for CRUD operations and analytics.

Frontend

- **HTML/CSS**:
    - Developed user-friendly forms and tables for data input and visualization.
    - Ensured responsive design for different screen sizes.

Security Measures

- Validated login IDs to prevent unauthorized access.
- Flash messages help users identify issues and correct errors in real-time.

Routes/Page Functionalities

1. Add Employee (`/add_employee`)

1. **Functionality**: This page allows administrators and coordinators to add a new employee to the system. Users must input all the required details into a form, including employee ID, name, role, phone number, department, and work email.
2. **Steps to Use**:
    - Navigate to the "Add Employee" page using the navigation bar.
    - Fill in the form fields:
        i. **Employee ID**: Enter a unique identifier for the employee.
        ii. **First Name, Middle Initial, Last Name**: Provide the full name of the employee.
        iii. **Role**: Specify the employee's role (e.g., worker, coordinator).
        iv. **Phone Number**: Add the employee's contact number.
        v. **Department**: Select the department (e.g., finance, sales, operations, or marketing).
        vi. **Work Email**: Enter a valid email address. (@gre.com)
    - Click the **Add Employee** button to submit the form.
    - If the employee ID is already in use, a flash message will notify the user.
    - Upon success, the new employee is added to the `employee` table, and if the role is "coordinator," additional details are inserted into the `wellness_coordinator` table.

2. Delete Employee (`/delete_employee`)

- **Purpose:** Allows users with "secretary" or "coordinator" credentials to delete an employee from the database.
- **Functionality:**
    - Users can enter an employee ID to delete.
    - Checks if the employee exists; if not, displays a flash message.
    - If the employee exists, deletes their record from the database.
    - Provides feedback on success or failure.

3. Enroll Employee (`/enroll_employee`)

- **Purpose:** Enables the enrollment of an employee into a wellness program.
- **Functionality:**
    - Takes input for employee ID and program ID.
        - **Valid Program ID's currently:** 201, 202, 203, 205.
    - **Validates:**

        - Employee and program existence.
        - Whether the employee is already enrolled in the program.
    - Enrolls the employee if all conditions are met.
    - Displays appropriate flash messages for success or errors.

4. Add Wellness Program (`/add_wellness_program`)

- **Purpose:** Allows the creation of a new wellness program.
- **Functionality:**
    - Requires input for program name, type, start date, end date, coordinator ID, and program ID.
    - **Validates:**
        - The uniqueness of the program ID.
        - The validity of the coordinator ID.
    - Adds the program to the database if validation passes.
    - Provides feedback via flash messages for success or failure.

5. View Health Metrics (`/view_health_metric`)

- **Purpose:** Displays the most recent health metrics of employees.
- **Functionality:**
    - Fetches the latest health metrics (e.g., BMI, blood pressure) for each employee.
    - Displays the data in descending order of the measurement date.
    - Handles errors gracefully and shows an empty table if no data is available.

6. Create Health Metric (`/create_health_metric`)

- **Purpose:** Allows users to add health metrics for employees.
- **Functionality:**
    - Accepts inputs such as employee ID, date measured, cholesterol levels, BMI, etc.
    - Validates the existence of the employee ID.
    - Inserts the data into the `health_metrics` table.
    - Provides feedback via flash messages for success or error handling.

7. View Enrollment List (`/view_enrollment_list`)

- **Purpose:** Displays a list of employees enrolled in wellness programs.
- **Functionality:**
    - On GET: Shows all enrollments.
    - On POST: Filters enrollments based on a specific program ID.
    - Displays enrollment details such as employee name, program ID, and program name.

8. View Department Breakdown (`/view_department_breakdown`)

- **Purpose:** Provides a summary of departments and the number of employees in each department.
- **Functionality:**
    - Fetches department names and employee counts.
    - Displays the data in a table.
    - Handles errors and displays a message if data is unavailable.

Database Design

Tables in the Database

1. **Employee Table**:
    - Attributes: ID, name, role, department, email.
2. **Wellness Coordinator Table**:
    - Attributes: Employee ID (FK), programs assigned.
3. **Programs Table**:
    - Attributes: Program ID, name, type, start date, end date, coordinator.

Summary

This system offers a comprehensive solution for managing wellness programs while ensuring data security and usability. By utilizing Flask for logic handling, MySQL for robust data management, and HTML/CSS for an interactive UI, the application fulfills its goal to improve employee wellness tracking and program administration effectively.