SAKECHA – FRANCHISEE PERFORMANCE TRACKER DOCUMENTATION

PostgreSQL

Username	postgres
Password	5539
Database	sakecha
Host	Localhost / 127.0.0.1
Port	5432

Server

Server Type	Flask
URL	http://127.0.0.1:5000/

Requirements

Flask==2.3.2
Flask-SQLAlchemy==3.0.3
Flask-Login==0.6.2
Werkzeug==2.3.4
pdfkit==1.0.0
psycopg2-binary==2.9.7
itsdangerous==2.1.2

.env FILE

```
# Flask app secret key
SECRET_KEY=your-very-secure-secret-key

# Database connection URL
DATABASE_URL=postgresql://postgres:5539@localhost/sakecha

# SMTP email server configuration
SMTP_SERVER=smtp.gmail.com
SMTP_PORT=587
SMTP_USERNAME= - @gmail.com
SMTP_PASSWORD= -
EMAIL_FROM= - (same as SMTP_USERNAME)@gmail.com
```

Steps to Use

1. Create the PostgreSQL database
   - Name: sakecha
   - Password: 5539
2. Install wkhtmltopdf from web
3. Terminal Steps:

```
cd SaaS_Proj
pip install -r requirements.txt
Set-ExecutionPolicy -Scope Process -ExecutionPolicy Bypass
python -m venv venv
venv\Scripts\activate
flask initdb
flask run
```

4. Website: http://127.0.0.1:5000/login

Login Testing

Dummy Accounts

Admin: admin | adminpass

Booth A Test: booth1 | pass1

Booth B Test: booth2 | pass2

Email Token

SMTP: Set from your own Gmail account

Folder Architecture

```
SaaS_Project/
├── app.py
├── templates/
    ├── login.html
    ├── register.html
    ├── forgot_password.html
    ├── reset_password.html
    ├── admin_dashboard.html
    ├── franchisee_dashboard.html
    ├── profile.html
    ├── users.html
    ├── sales_entry_form.html
    ├── attendance_log_form.html
    ├── ingredient_reorder_form.html
    ├── admin_sales_page.html
    └── report.html
```

Database Models

User:

- Represents users of the application.

- Attributes: **id**, **username**, **email**, **password_hash**, **role**, **booth_id**, **approved**, **reset_token**, **reset_token_expiration**.

Booth:

- Represents booths where users (franchisees) operate.
- Attributes: **id**, **name**, **location**.
- Relationship: One-to-many with **User**.

SalesEntry:

- Represents sales entries made by users.
- Attributes: **id**, **user_id**, **date**, **drink_name**, **quantity**, **booth_id**.
- Relationship: Many-to-one with **User** and **Booth**.

AttendanceLog:

- Represents attendance logs for users.
- Attributes: **id**, **user_id**, **username**, **date**, **present**, **booth_id**.
- Relationship: Many-to-one with **Booth**.

IngredientReorder:

- Represents requests for ingredient reorders.
- Attributes: **id**, **user_id**, **username**, **booth_id**, **date**, **ingredient_name**, **quantity**, **status**.
- Relationship: Many-to-one with **Booth**.

BoothChangeRequest:

- Represents requests for booth changes by users.
- Attributes: **id**, **user_id**, **requested_booth_id**, **status**, **created_at**.
- Relationship: Many-to-one with **User** and **Booth**.

Application Route

/login

- Method: GET, POST
- Description: Handles user login. If the user is already authenticated, redirects to the dashboard. On POST, validates credentials and logs in the user if approved.

/register

- Method: GET, POST
- Description: Allows new users to register. On POST, checks for existing usernames/emails, creates a new user with an unapproved status, and assigns a booth if provided.

/approve_user

- Method: POST
- Description: Admin route to approve a user account. Requires admin role and user ID in the request body.

/reject_user

- Method: POST
- Description: Admin route to reject and delete a user account. Requires admin role and user ID in the request body.

/forgot_password

- Method: GET, POST

- Description: Allows users to request a password reset. On POST, generates a reset token and sends it via email.

/reset_password

- Method: GET, POST
- Description: Allows users to reset their password using a token. Validates the token and updates the password if valid.

/profile

- Method: GET, POST
- Description: Displays and allows users to update their profile information, including password changes.

/logout

- Method: GET
- Description: Logs out the current user and redirects to the login page.

/dashboard

- Method: GET
- Description: Displays the dashboard for the logged-in user. Admins see sales summaries and pending approvals, while franchisees see their sales and attendance.

/sales_entry

- Method: GET, POST
- Description: Allows franchisees to enter sales data. Validates input and saves sales entries to the database.

/attendance_log

- Method: GET, POST
- Description: Allows users to log their attendance. Validates input and saves attendance records.

/ingredient_reorder

- Method: GET, POST
- Description: Allows users to submit ingredient reorder requests. Validates input and saves the request.

/update_ingredient_status

- Method: POST
- Description: Admin route to update the status of an ingredient reorder request. Requires admin role and request ID in the request body.

/delete_ingredient_request

- Method: POST
- Description: Admin route to delete an ingredient reorder request. Requires admin role and request ID in the request body.

/request_booth_change

- Method: POST

- Description: Allows franchisees to request a change of booth. Validates booth selection and checks for existing requests.

/approve_booth_change

- Method: POST
- Description: Admin route to approve a booth change request. Requires admin role and request ID in the request body.

/reject_booth_change

- Method: POST
- Description: Admin route to reject a booth change request. Requires admin role and request ID in the request body.

/generate_report

- Method: GET
- Description: Admin route to generate a monthly report of sales, attendance, and ingredient reorders. Returns a PDF file.

/users

- Method: GET, POST
- Description: Admin route to manage users. On POST, handles updates to user assignments, booth locations, and booth creation.

/delete_user

- Method: POST
- Description: Admin route to delete a user account. Requires admin role and user ID in the request body.

/delete_booth

- Method: POST
- Description: Admin route to delete a booth. Checks for existing sales entries before deletion. Requires admin role and booth ID in the request body.

/sales

- Method: GET
- Description: Admin route to view sales data for the last month. Displays aggregated sales information.

/initdb

- Method: CLI Command
- Description: Initializes the database by dropping existing tables, creating new ones, and populating them with demo data.