**Password Manager**

A simple password management application built with Flask, Docker, and Docker Compose. This application allows you to add, retrieve, and list passwords securely.

**Table of Contents**

1. [Features](#features)
2. [Prerequisites](#prerequisites)
3. [Setup](#setup)
4. [Usage](#usage)
5. [Docker Instructions](#docker-instructions)
6. [File Structure](#file-structure)
7. [License](#license)

**Features**

* Add passwords for various services
* Retrieve passwords for specific services
* List all stored passwords
* Secure storage of passwords using encryption

**Prerequisites**

Before you start, ensure you have the following installed:

* Python 3.11+
* Docker and Docker Compose
* Git

**Setup**

1. **Clone the Repository**

bash

Copy code

git clone https://github.com/Olisaemeka111/password-project.git

cd password-project

1. **Install Dependencies**

If you prefer to set up the environment manually (not using Docker):

bash

Copy code

pip install -r requirements.txt

Create a .env file if necessary and ensure your environment variables are set up.

**Usage**

1. **Run Locally**

To run the Flask application locally:

bash

Copy code

flask run

By default, it will be available at http://127.0.0.1:5000.

1. **Endpoints**
   * POST /add\_password - Add a password for a service
     + Form fields: service, master\_password
   * POST /get\_password - Retrieve a password for a service
     + Form fields: service, master\_password
   * POST /list\_passwords - List all stored passwords
     + Form field: master\_password

**Docker Instructions**

1. **Build the Docker Image**

bash

Copy code

docker build -t my\_flask\_app .

1. **Run the Docker Container**

bash

Copy code

docker run -p 80:80 my\_flask\_app

The application will be accessible at http://localhost:80.

1. **Using Docker Compose**

If you have docker-compose.yml, you can build and run the container with:

bash

Copy code

docker-compose up

This will automatically build the image (if not already built) and run the container.

**File Structure**

* app.py - The main Flask application file.
* Dockerfile - Dockerfile for creating the Docker image.
* docker-compose.yml - Docker Compose configuration file.
* requirements.txt - Python dependencies.
* templates/ - Directory containing HTML templates.
* static/ - Directory containing static files like CSS, JS, images.
* salt.txt - File to store the salt for encryption.
* passwords.enc - Encrypted file to store passwords.

**License**

This project is licensed under the MIT License - see the LICENSE file for details.

**Associated Codes in the Directory**

Here's a brief overview of the associated codes and their purpose:

1. **app.py** - Main application file containing Flask routes and functions for managing passwords.
2. **Dockerfile** - Defines the Docker image build process for the Flask application.
3. **docker-compose.yml** - Configures services, networks, and volumes for Docker Compose.
4. **requirements.txt** - Lists Python packages needed to run the application.
5. **templates/index.html** - HTML template for the main page.
6. **static/videos/istockphoto-1255754954-640\_adpp\_is.mp4** - Background video for the main page.
7. **static/images/vecteezy\_banner-background-full-color-blue-and-black-gradations\_17396302.jpg** - Background image for JSON output styling.