Step Book

Pin Food Web App Flask

Episode 01

About Project . Preparation . Database Setup

What is “Step Book”?

Step Book is a book that contains the steps of how to build something. Does not contain a detailed description in it. This book is specially designed to focus on practical coding to make a real project.

How to use “Step Book”?

This is an important part of how I will show you how to use this book.

Since this book is only step by step, I recommend following it. You will find a point where you will ask "what does that mean blah-blah?". search in the Google search field. This step is much more practical and quicker to learn something. You continue to build a project, and when you get confused, you immediately find out with the help of Google or official documentation.

In this book, you will practice how to **improve your project building skills**, **critical thinking skills**, and **problem solving skills**. That's the point. If you're still confused about the steps, go back to Videos on our YouTube channel.

Source Code

If any time you need our source code, we have already provided it on our GitHub. Visit the following link:

https://github.com/try-fullstack/pin-food-web-app-flask

Support us at:

- **YouTube**
  https://www.youtube.com/channel/UCXLNeTmZoyxN1Bb5LI7zS2Q
- **Patreon**
  https://www.patreon.com/tryfullstack

Sorry, if our English is not perfect. We'll keep trying to make it easier to read.

TRY FULL STACK

Table of Contents

About This Project

Is a web application that displays a page containing an interactive map. The user can mark the location of a new "food stall" on the map. Users can enter the date, category, and description of the "food stall".

Users can also see places for food stalls that have been marked before by displaying an icon on the map and if the user hovers the mouse over an icon, a more complete description of the food stall will appear.

The goal is, users can easily see which areas have food stalls according to category and help Vlogers come to these places to create their own YouTube content.

Preparation

Open a terminal or command line, then enter into any drive (for example, drive D if you are using windows). Create a folder called **pinfood**:

```
mkdir pinfood
cd pinfood
```

Create a virtual environment named **.venv**:

```
python -m venv .venv
.venv\Scripts\activate
```

Install the **PyMySQL** and **Flask** packages. PyMySQL is used to communicate with MySQL:

```
pip install flask pymysql
```

Database Setup

Inside the project root, create a file called **dbconf.py** and add the following configuration:

```
test = True

dbuser = 'root'
dbpassword = ''
```

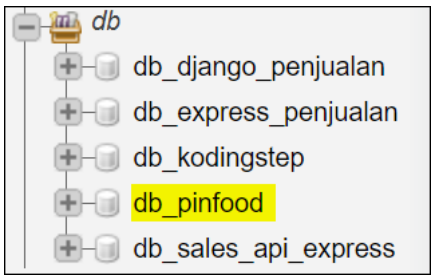
Let's create a file called **dbsetup.py** inside the project root. Then, type the following code:

```
import pymysql
import dbconf

connection = pymysql.connect(host='localhost',
                             user=dbconf.dbuser,
                             passwd=dbconf.dbpassword)

try:
    with connection.cursor() as cursor:
        sql = "CREATE DATABASE IF NOT EXISTS db_pinfood"
        cursor.execute(sql)

        sql = """CREATE TABLE IF NOT EXISTS db_pinfood.foods (
                id int NOT NULL AUTO_INCREMENT,
                latitude FLOAT(10,6),
                longitude FLOAT(10,6),
                date DATETIME,
                category VARCHAR(50),
                description VARCHAR(1000),
                updated_at TIMESTAMP,
                PRIMARY KEY (id)
                )"""

        cursor.execute(sql)

    connection.commit()

finally:
    connection.close()
```

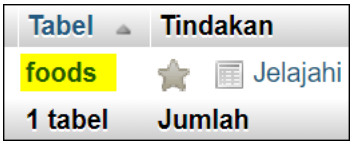
Let's turn on **MySQL Server**. Then, run **dbsetup.py** to create database and table automatically:

```
python dbsetup.py
```

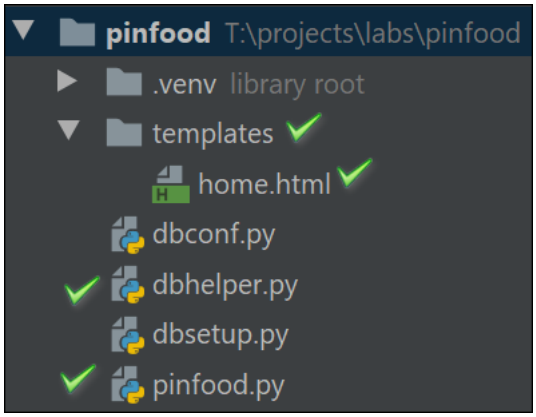
Make sure the database is created successfully by viewing it in the **MySQL console** using **SHOW DATABASES;** or you can use **PHPMyAdmin**.



Also, make sure the **foods** table in the **db_pinfood** is created as well:



Create a directory structure and files like:



Let's open the **dbhelper.py** file and add the following code:

```
import pymysql
import dbconf


class DBHelper:

    def connect(self, database="db_pinfood"):
        return pymysql.connect(host='localhost',
                               user=dbconf.dbuser,
                               passwd=dbconf.dbpassword,
                               db=database)

    def get_all_inputs(self):
        connection = self.connect()

        try:
```

```
            query = "SELECT description FROM foods;"
            with connection.cursor() as cursor:
                cursor.execute(query)
            return cursor.fetchall()

        finally:
            connection.close()

    def add_input(self, data):
        connection = self.connect()
        try:
            # The following introduces a deliberate security
            # flaw. See section on SQL injection below
            query = "INSERT INTO foods " \
                    "(description) VALUES " \
                    "('{}');".format(data)
            with connection.cursor() as cursor:
                cursor.execute(query)
                connection.commit()

        finally:
            connection.close()

    def clear_all(self):
        connection = self.connect()

        try:
            query = "DELETE FROM foods;"
            with connection.cursor() as cursor:
                cursor.execute(query)
                connection.commit()
        finally:
            connection.close()
```

Next, open the **pinfood.py** file and add the following code:

```
from dbhelper import DBHelper
from flask import Flask
from flask import render_template
from flask import request

app = Flask(__name__)
DB = DBHelper()


@app.route("/")
def home():
    try:
        data = DB.get_all_inputs()
    except Exception as e:
        print(e)
        data = None
    return render_template("home.html", data=data)


@app.route("/add", methods=["POST"])
def add():
    try:
```

```
        data = request.form.get("description")
        DB.add_input(data)
    except Exception as e:
        print(e)
    return home()


@app.route("/clear")
def clear():
    try:
        DB.clear_all()

    except Exception as e:
        print(e)

    return home()


if __name__ == '__main__':
    app.run(port=5000, debug=True)
```

Open the **templates/home.html** file and add the following code:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Pin Food</title>
  </head>
  <body>
    <div class="container mt-4 mb-3">
      <div class="col-md-12">
        <h1 class="display-4">Pin Food</h1>
        <hr>
      </div>
    </div>
    <div class="container mb-3">
      <div class="col-md-4">
        <form action="/add" method="POST">
          <label>Description</label>
          <input type="text" class="form-control" name="description">
          <button class="btn btn-primary" type="submit">
            Submit
          </button>
          <a href="/clear" class="btn btn-outline-danger">
            Clear
          </a>
        </form>
        <div class="mt-3">
          {% if data %}
            {% for description in data %}
              <p>{{ description }}</p>
            {% endfor %}
          {% endif %}
        </div>
      </div>
    </div>
  </body>
</html>
```

Run the project:

```
python pinfood.py
```

Open **http://localhost:5000** in a browser, you will see something like this. Try typing anything (for example, **test**) then click the **Submit** button:

In the **foods** table, the data should already be saved:

id	latitude	longitude	date	category	description	updated_at
4	*NULL*	*NULL*	*NULL*	*NULL*	Test description	2021-01-24 12:23:56
5	*NULL*	*NULL*	*NULL*	*NULL*	Test description again	2021-01-24 12:24:01
6	*NULL*	*NULL*	*NULL*	*NULL*	Test desc food again	2021-01-24 12:24:06