

Name:Sanika Santosh Deshmukh

Domain:Python

Code:

```
import os
import sqlite3
import string
import random
from datetime import datetime

from flask import Flask, request, redirect, url_for, render_template_string, abort

DB_NAME = "urls.db"
SHORT_CODE_LENGTH = 6

app = Flask(__name__)
def get_db_connection():
    """Create a connection to the SQLite database."""
    conn = sqlite3.connect(DB_NAME)
    conn.row_factory = sqlite3.Row
    return conn
def init_db():
    """Initialize the database with the required table."""
    conn = get_db_connection()
    cur = conn.cursor()
    cur.execute(
        """
        CREATE TABLE IF NOT EXISTS urls (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            short_code TEXT UNIQUE NOT NULL,
            original_url TEXT NOT NULL,
```

```

        created_at TEXT NOT NULL

    );

"""

)

conn.commit()

conn.close()

```

```

def generate_short_code(length: int = SHORT_CODE_LENGTH) -> str:
    """Generate a random short code consisting of letters and digits."""
    characters = string.ascii_letters + string.digits
    return "".join(random.choice(characters) for _ in range(length))

```

```

def create_short_url(original_url: str) -> str:
    """
    Create a new short URL mapping in the database.

    Ensures the generated short code is unique by checking the database.
    """
    conn = get_db_connection()
    cur = conn.cursor()

```

```

    while True:
        short_code = generate_short_code()
        cur.execute("SELECT 1 FROM urls WHERE short_code = ?", (short_code,))
        if cur.fetchone() is None:
            break

```

```

        created_at = datetime.utcnow().isoformat()
        cur.execute(
            "INSERT INTO urls (short_code, original_url, created_at) VALUES (?, ?, ?);",
            (short_code, original_url, created_at),

```

```
)  
conn.commit()  
conn.close()  
return short_code
```

```
def get_original_url(short_code: str) -> str | None:  
    """Retrieve the original URL for a given short code."""  
    conn = get_db_connection()  
    cur = conn.cursor()  
    cur.execute("SELECT original_url FROM urls WHERE short_code = ?;", (short_code,))  
    row = cur.fetchone()  
    conn.close()  
    if row:  
        return row["original_url"]  
    return None
```

```
INDEX_TEMPLATE = """  
<!doctype html>  
<html lang="en">  
    <head>  
        <meta charset="utf-8">  
        <title>URL Shortener</title>  
        <style>  
            body {  
                font-family: Arial, sans-serif;  
                background-color: #f4f4f9;  
                display: flex;  
                justify-content: center;  
                align-items: center;  
                height: 100vh;  
                margin: 0;
```

```
}  
  
.container {  
  background: #ffffff;  
  padding: 24px 32px;  
  border-radius: 12px;  
  box-shadow: 0 4px 12px rgba(0, 0, 0, 0.08);  
  width: 100%;  
  max-width: 480px;  
}  
  
h1 {  
  margin-top: 0;  
  text-align: center;  
  color: #333333;  
}  
  
form {  
  display: flex;  
  flex-direction: column;  
  gap: 12px;  
  margin-top: 16px;  
}  
  
input[type="url"] {  
  padding: 10px 12px;  
  border-radius: 8px;  
  border: 1px solid #cccccc;  
  font-size: 14px;  
}  
  
button {  
  padding: 10px 12px;  
  border-radius: 8px;  
  border: none;  
  background-color: #4a90e2;
```

```
    color: #ffffff;

    font-size: 14px;

    cursor: pointer;
}

button:hover {

    background-color: #357ab8;
}

.result {

    margin-top: 16px;

    padding: 10px 12px;

    background-color: #f0f7ff;

    border-radius: 8px;

    font-size: 14px;

    word-break: break-all;
}

.error {

    margin-top: 16px;

    padding: 10px 12px;

    background-color: #ffe6e6;

    border-radius: 8px;

    color: #b00020;

    font-size: 14px;
}

small {

    display: block;

    margin-top: 8px;

    color: #777777;
}

</style>

</head>

<body>
```

```

<div class="container">

    <h1>URL Shortener</h1>

    <form method="post" action="{{ url_for('shorten') }}">

        <label for="url">Enter a long URL:</label>

        <input id="url" type="url" name="url" placeholder="https://example.com/very/long/link"
required>

        <button type="submit">Shorten URL</button>

        <small>Example: paste any long URL, we will return a shorter link.</small>

    </form>

    {% if short_url %}

    <div class="result">

        <strong>Short URL:</strong>

        <a href="{{ short_url }}" target="_blank">{{ short_url }}</a>

    </div>

    {% endif %}

    {% if error %}

    <div class="error">

        {{ error }}

    </div>

    {% endif %}

</div>

</body>

</html>

"""

```

```

@app.route("/", methods=["GET"])

```

```

def index():

```

```

    """Render the main page with the URL input form."""

```

```

    return render_template_string(
INDEX_TEMPLATE, short_url=None, error=None)

```

```

@app.route("/shorten", methods=["POST"])
def shorten():
    """Handle form submission, create short URL, and show result."""
    original_url = request.form.get("url", "").strip()

    if not original_url:
        return render_template_string(
            INDEX_TEMPLATE, short_url=None, error="Please provide a valid URL."
        )

    if not original_url.startswith(("http://", "https://")):
        original_url = "http://" + original_url

    short_code = create_short_url(original_url)
    short_url = url_for("redirect_short_url", short_code=short_code, _external=True)

    return render_template_string(INDEX_TEMPLATE, short_url=short_url, error=None)

@app.route("/<short_code>")
def redirect_short_url(short_code: str):
    """Redirect a short code to the original URL."""
    original_url = get_original_url(short_code)
    if original_url is None:
        abort(404)
    return redirect(original_url)

if __name__ == "__main__":
    init_db()
    port = int(os.environ.get("PORT", 5000))
    app.run(debug=True, port=port)

```

Output:

URL Shortener

Enter a long URL:

<https://www.getintocanva.com>

Shorten URL

Example: paste any long URL, we will return a shorter link.

Short URL: <http://127.0.0.1:5000/2dopJN>