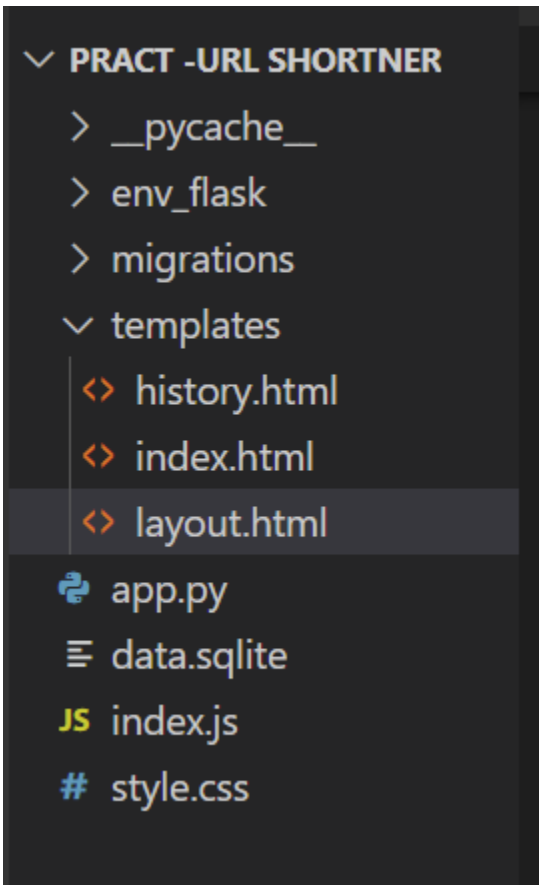
URL Shortener Web Application

Directory Structure



Importing the Required libraries

- Flask: for backend app
- render_template:for rendering the html files
- request:to take the input passed by user on frontend to backend
- redirect: To redirect one page to another
- url_for:to directly jump to route using the function name
- flash: to give some warning messages if user passed any invalid url

- Os: to use the file system using python code

```
from flask import Flask,render_template,request,redirect,url_for,flash
import os
from random import choice
import string
from flask_sqlalchemy import SQLAlchemy
from flask_migrate import Migrate
```

 # A secret key is generated using os module

```
app.secret_key = os.urandom(24)
```

A unique key is setted so the session remains available

 # SQLAlchemy Configuration and pass the application into SQLAlchemy class

```
basedir = os.path.abspath(os.path.dirname(__file__))
path = 'sqlite:///' + os.path.join(basedir, 'data.sqlite')
app.config['SQLALCHEMY_DATABASE_URI'] = path
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

db = SQLAlchemy(app)
Migrate(app, db)
```

 # creating model/table

I have created a class named ShortUrls and table name is shorturls
In this table I have 4 columns :

1. id
2. original_url
3. short_id
4. short_url

```
class ShortUrls(db.Model):
    __tablename__ = 'shorturls'
    id = db.Column(db.Integer,primary_key = True)
    original_url = db.Column(db.String(500), nullable=False)
    short_id = db.Column(db.String(20), nullable=False, unique=True)
    short_url = db.Column(db.String(500))


    def __init__(self, original_url,short_id,short_url):
        self.original_url = original_url
        self.short_id = short_id
        self.short_url = short_url

    def __repr__(self):
        return "original_url - {} short_id - {} short_url -
{}".format(self.original_url, self.short_id,self.short_url)
```

Function to generate unique id

```
def generate_short_id(num_of_chars: int):

    """Function to generate short_id of specified number of characters"""

    return ''.join(choice(string.ascii_letters+string.digits) for _ in
range(num_of_chars))
```

This code generated a unique token which later used in short URL

Static Routes

```python
@app.route('/', methods=['GET', 'POST'])
def index():
    if request.method == 'POST':
        short_id = generate_short_id(8)
        url = request.form['url']
        short_url = request.host_url + short_id
        new_link = ShortUrls(original_url=url, short_id=short_id,short_url=short_url)
        db.session.add(new_link)
        db.session.commit()
        print("data added................")
        return render_template('index.html', short_url=short_url)

    return render_template('index.html')


@app.route('/history')
def history():
    records = ShortUrls.query.all()
    return render_template('history.html',records=records)
```

In the above code I have fetched the url link provided by user and create a unique token which later got combined and stored in the sqlite database

Dynamic Routes

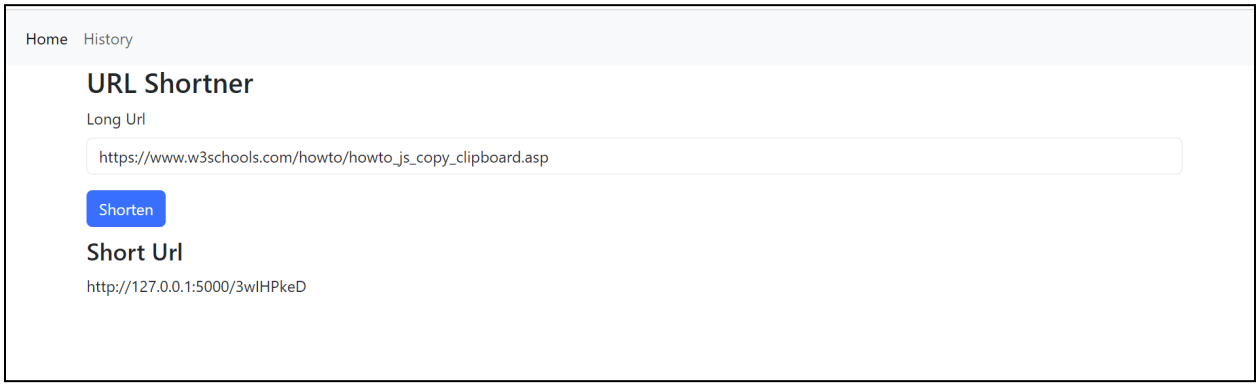
```python
@app.route('/<short_id>')
def redirect_url(short_id):
    link = ShortUrls.query.filter_by(short_id=short_id).first()
    if link:
        return redirect(link.original_url)
    else:
        flash('Invalid URL')
        return redirect(url_for('index'))
```

When user enter the short url in browser so the dynamic routes takes the value of shortid and matches in the database if id is present or not. If id is present so the browser takes the original link values from db and redirect it to original link whereas if id doesn’t matches with database then a flash message is generated as Invalid Url

Frontend Look

Home Page

On entering Url



History Page

Home History

Original URL	Short Id	Short URL
	t2yi2Fat	None
https://github.com/bansalkanav/Machine_Learning_and_Deep_Learning/blob/master/Module%201%20-%20Python%20Programming/09.%20Databases/SQLAlchemy%20ORM.ipynb	3AwzbRfe	None
https://github.com/exsquared/class-of-2023	i4TaGqpD	http://127.0.0.1:5000/i4TaGqpD
https://github.com/exsquared/class-of-2023	BFG7kHlZ	http://127.0.0.1:5000/BFG7kHlZ
https://github.com/bansalkanav/Machine_Learning_and_Deep_Learning/blob/master/Module%201%20-%20Python%20Programming/09.%20Databases/SQLAlchemy%20ORM.ipynb	YYoMIHaE	http://127.0.0.1:5000/YYoMIHaE
https://www.google.com/	VvDEOmaG	http://127.0.0.1:5000/VvDEOmaG
https://getbootstrap.com/docs/5.3/content/tables/#overview	Bec55C5B	http://127.0.0.1:5000/Bec55C5B
https://getbootstrap.com/docs/5.3/content/tables/#overview	aPs2scph	http://127.0.0.1:5000/aPs2scph
https://getbootstrap.com/docs/5.3/content/tables/#overview	csQOkfOt	http://127.0.0.1:5000/csQOkfOt
https://getbootstrap.com/docs/5.3/content/tables/#overview	zoNP8N86	http://127.0.0.1:5000/zoNP8N86
https://getbootstrap.com/docs/5.3/content/tables/#overview	KfHEPTx6	http://127.0.0.1:5000/KfHEPTx6
https://getbootstrap.com/docs/5.3/content/tables/#overview	ria27sGL	http://127.0.0.1:5000/ria27sGL
https://getbootstrap.com/docs/5.3/content/tables/#overview	VMowvKzZ	http://127.0.0.1:5000/VMowvKzZ