Url Shortner Web Application

Objective of Project:

1. As the name suggests, it shortens URLs.
2. Users can also save URLS by coming to the web app.

Why do we need URL Shortener?

1. Sometimes we need to share or send links and this can be tiresome and annoying to
2. copy and paste long URLs. That is where URL shorteners come in. Not only it helps in
3. shortening the URL but it also allows the user to copy the shortened URL with a click of a button.

What this project Include:

The project consists of 2 parts:

1. Frontend
   a. HTML,
   b. CSS
   c. Bootstrap

1. Backend

   a. Backend Server - Flask

   b. Backend Database- ORM

Project Procedure:

1. Frontend:

A. HTML templates

Here I have created 4 HTML pages

1. **layout.html**
2. **short_url.html**
3. **search.html**
4. **history.html**

1. layout.html:

- This html page simply a home page for my URL Shortner Web Application.

2. short_url.html

- This page contained two input tags one for **URL** and another one for **Title.**

- Short url Button and **Copy Button.** To create Copy button I used Script.

- All the Tags included in the **Table**.

3. search.html

- This page accept the URL to be searched and if the URL is present in the Database, Render that URL in the Table.

4. history.html

- This page renders all the previously shortned URL in the table form.

B. CSS

- **Used the CSS to style the HTML pages, Tables, Buttons, Warning**

Message Boxes.

2. Backend

A. Backend Web Server:

1. Need to install and import all the required packages for creating servers, packages are
   Flask, request, render_template, url_for etc
   For Validating URL - validators package
   For URL Shortning- pyshortners package

2. Created a object of class Flask
   app=Flask(__name__)

3. Created a Multiple Routes or end-points,
   a. One for Home route
   b. Url Shortner route
   c. Search route
   d. History route

   a. Home route
   - Home page I just rendered a Home Page

   b. URL Shortner route.
   - Creaed two request methods **POST and GET.**
   - For the **Request=POST**, Following steps will execute,
     1. Created two Input variable to accept the required parameters **1. URL 2. title for URL**
     2. Here I have checked whether the entered **URL is a valid URL or NOT** using package called **validators.**
     3. If the entered URL is valid then only my program going to create short url, here I used **pyshorteners package,** In pyshorteners I used **tinyurl.com** as a third-party server to short the url.
     4. Finally **Rendering the URL Shortner Page in the HTML template.**

- For **Request=GET,** Render the **the URL Shortner Page in the HTML template with No parameters.**

c. Search route:
- If the user enter the search end point, here user search for previously shortned URL.

d. History route:
- If the user enter the history end point, they see all the previously shortned URL's.

4. Instance to run the flask app
```
if __name__=='__main__':
   app.run(debug=True)
```

B. Backend Database

1. Here the backend Database connection is called ORM(Object Relational Mapper)
2. Imported the required libraries to create Database **SQLAlchemy** and **MIgrate.**
3. Create a database object and pass the application into it.
4. Create a table and columns in the Database.
   - Table with name **short_urls**
   - Columns in the table are **ID, Original url, Title and Shotned url.**

   Note: After creating table and columns in Flask backend server go to command promt and activate flask environment and run following command,
   ```
   * flask db init
   * flask db migrate -m "First Migration"
   * flask db upgrade
   ```

5. Following are the **actions take place in each route**,

a. In the Short URL route,

- Two inputs URL and Title are stored in the database using the query
URL_Shortner(url, title, short_url)

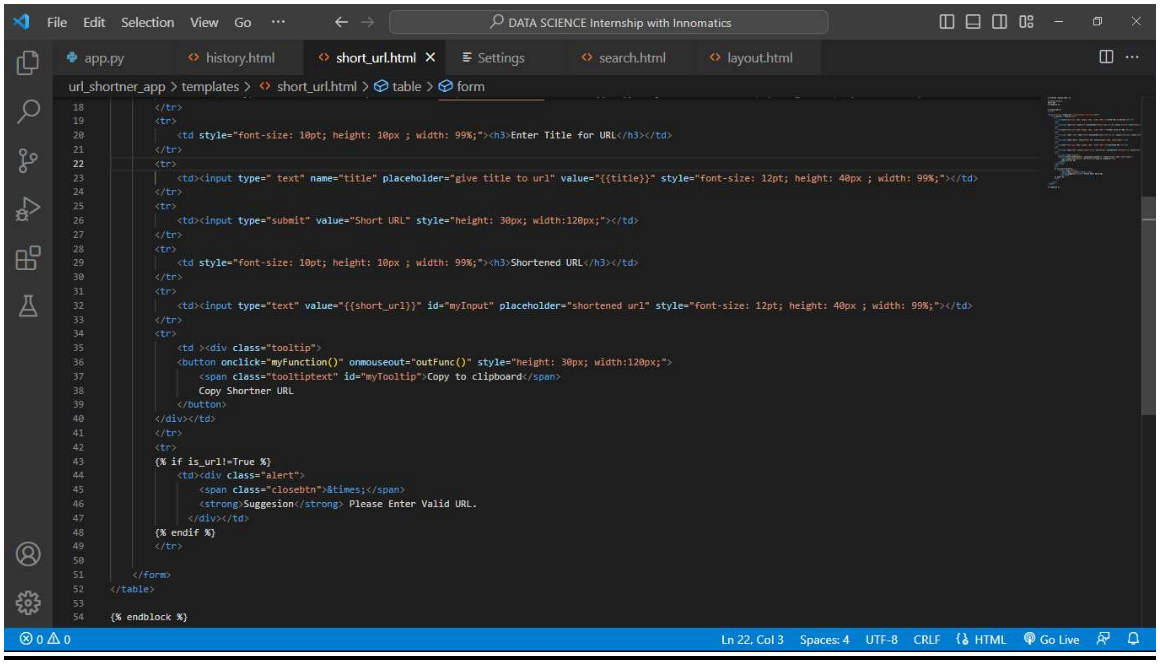
b. In the Search URL route,

- The url to be searched is accepted from the user and the following query is to executed,
URL_Shortner.query.filter_by(org_url=url).first()

c. In the History route,

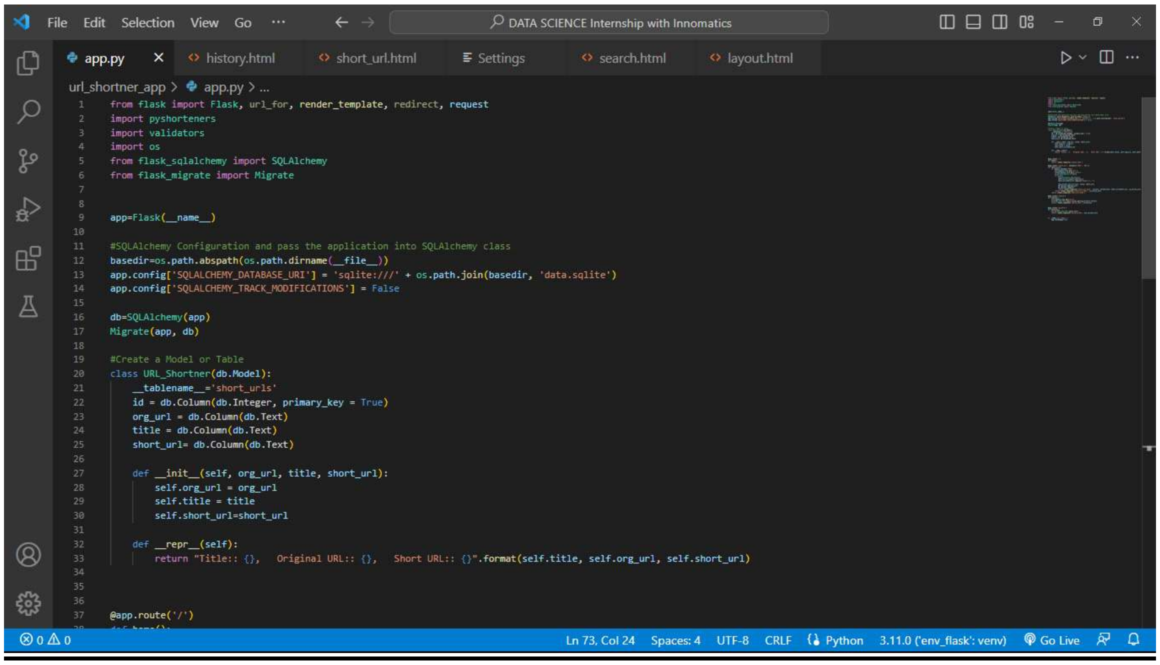
- URL_Shortner.query.all() this query is executed to list out all the previosly shortned url.

HTML templates code.


File Edit Selection View Go ···
DATA SCIENCE Internship with Innomatics
app.py
history.html
short_url.html
Settings
search.html
layout.html
url_shortner_app > templates > short_url.html > table > form
Ln 22, Col 3 Spaces: 4 UTF-8 CRLF HTML Go Live

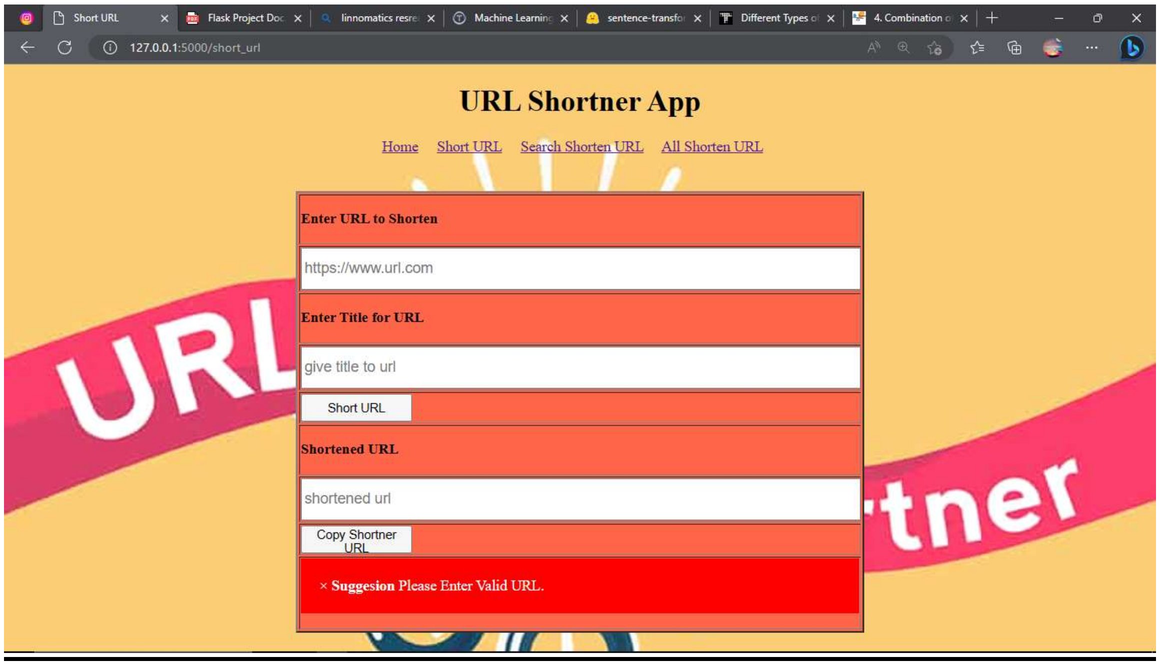
```html
        </tr>
        <tr>
            <td style="font-size: 10pt; height: 10px ; width: 99%;"><h3>Enter Title for URL</h3></td>
        </tr>
        <tr>
            <td><input type=" text" name="title" placeholder="give title to url" value="{{title}}" style="font-size: 12pt; height: 40px ; width: 99%;"></td>
        </tr>
        <tr>
            <td><input type="submit" value="Short URL" style="height: 30px; width:120px;"></td>
        </tr>
        <tr>
            <td style="font-size: 10pt; height: 10px ; width: 99%;"><h3>Shortened URL</h3></td>
        </tr>
        <tr>
            <td><input type="text" value="{{short_url}}" id="myInput" placeholder="shortened url" style="font-size: 12pt; height: 40px ; width: 99%;"></td>
        </tr>
        <tr>
            <td ><div class="tooltip">
            <button onclick="myFunction()" onmouseout="outFunc()" style="height: 30px; width:120px;">
                <span class="tooltiptext" id="myTooltip">Copy to clipboard</span>
                Copy Shortner URL
            </button>
        </div></td>
        </tr>
        <tr>
        {% if is_url!=True %}
            <td><div class="alert">
                <span class="closebtn">&times;</span>
                <strong>Suggesion</strong> Please Enter Valid URL.
              </div></td>
        {% endif %}
        </tr>

    </form>
</table>

{% endblock %}
```

Backend Flask Code:


File Edit Selection View Go ···
DATA SCIENCE Internship with Innomatics
app.py
history.html
short_url.html
Settings
search.html
layout.html
url_shortner_app > app.py > ...
Ln 73, Col 24 Spaces: 4 UTF-8 CRLF Python 3.11.0 ('env_flask': venv) Go Live

```python
from flask import Flask, url_for, render_template, redirect, request
import pyshorteners
import validators
import os
from flask_sqlalchemy import SQLAlchemy
from flask_migrate import Migrate


app=Flask(__name__)

#SQLAlchemy Configuration and pass the application into SQLAlchemy class
basedir=os.path.abspath(os.path.dirname(__file__))
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///' + os.path.join(basedir, 'data.sqlite')
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

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

#Create a Model or Table
class URL_Shortner(db.Model):
    __tablename__='short_urls'
    id = db.Column(db.Integer, primary_key = True)
    org_url = db.Column(db.Text)
    title = db.Column(db.Text)
    short_url= db.Column(db.Text)

    def __init__(self, org_url, title, short_url):
        self.org_url = org_url
        self.title = title
        self.short_url=short_url

    def __repr__(self):
        return "Title:: {},  Original URL:: {},  Short URL:: {}".format(self.title, self.org_url, self.short_url)



@app.route('/')
```

Running application Tab



URL Shortner App

Home Short URL Search Shorten URL All Shorten URL

Enter URL to Shorten

https://www.url.com

Enter Title for URL

give title to url

Short URL

Shortened URL

shortened url

Copy Shortner URL

× **Suggesion** Please Enter Valid URL.