**URL Shortener Web Application (Advance)**

**Project Workflow**

1. During signup, the user has to enter a unique username. If he enters a username which is already taken then an error message will occur displaying “This username already exists…” Then the user needs to enter another username. The length of the username is valid only in the range of 5 and 9. If the username is not of valid length, then an error message will occur: “Username must be between 5 to 9 characters long”.
2. After logging in, the main web-application opens which is the URL shortener. Here the user can enter the URL he wants to shorten. After entering a URL, click on the ‘shorten’ URL button to display the shortened URL in the following text-field which can be copied by clicking on the copy button.
3. After the ‘shorten’ button is clicked, the URL that is entered is saved in our database with the shortened URL. It is saved in the database so that the user can look into the previous URLs he entered in our web-app with their shortened URL in the forms given below the copy button.

Step-1

Install Necessary Packages

pip install flask

pip install flask-sqlalchemy

pip install flask-migrate

pip install validators

Step-2

Import the Packages

import os

from flask import Flask, render\_template, request, redirect, url\_for

from flask\_sqlalchemy import SQLAlchemy

from flask\_migrate import Migrate

from flask\_login import LoginManager, UserMixin, login\_required, login\_user, logout\_user

from werkzeug.security import generate\_password\_hash, check\_password\_hash

import string

import random

import validators

Step-3

Configure the SQL Alchemy

Next Create two Tables in Database

Urls- for storing Original url, full url and short url

Original and full url -string with size upto 500

Short url – string with size upto 10

Users- store user name password

email -string with size upto 64

password– string with size upto 128

Step-4

Create tables

@app.before\_first\_request

def create\_tables():

    db.create\_all()

Step-5

From flask\_login create login manager

login\_manager = LoginManager()

login\_manager.init\_app(app)

# Tell users what view to go to when they need to login.

login\_manager.login\_view = "login"

load the user

@login\_manager.user\_loader

def load\_user(user\_id):

    return User.query.get(user\_id)

Step-6

Take url from the user and validate using validators function

Pass url to shorten\_url() function

def shorten\_url():

    chars = string.ascii\_letters + string.digits

    while True:

        short\_url = ''.join(random.choice(chars) for \_ in range(6))

        full\_url = request.host\_url + short\_url

        if not Url.query.filter\_by(short\_url=short\_url).first():

            return short\_url,full\_url

1. The **shorten\_url** function takes the original URL as input and returns the shortened URL.
2. Generating a Unique String: The **chars** variable contains a string of all uppercase letters, lowercase letters, and digits. Using this, we can generate a unique string of 6 characters using the **random.choice** function.
3. Encoding the URL: The original URL is encoded using base64 encoding, which produces a string of characters that can be easily stored and retrieved from the database.
4. Saving to the Database: The original URL and the shortened URL can be saved to a database using an ORM or other database technology.
5. Returning the Shortened URL: The function returns the shortened URL, which includes the unique string and the domain name.

@app.route('/<short\_url>')

def redirect\_to\_original\_url(short\_url):

    url = Url.query.filter\_by(short\_url=short\_url).first()

    return redirect(url.original\_url)

Step-7

Ask user to login or register in order to shorten the url

 {% if current\_user.is\_authenticated %}

Then only go to shorten url page

Step-8

Html Pages

1. Home Page: The home page contains a form that allows the user to enter the original URL.
2. History Page: The history page displays a table of all the previously shortened URLs.
3. Index Page: Shows the Description of the Page
4. Layout Page: Consists of all CSS styles, Nav bar for all Pages
5. Login Page: Authentication in order to Shorten Link
6. Register Page: For Creating a user account
7. Result Page: Give Shorten URL to User for Original URL and Copy Button to Copy the Shorten Link

Step-9

Migrate the database to the app

Run these commands to initialize the Database

flask db init

flask db migrate -m “—Message--”

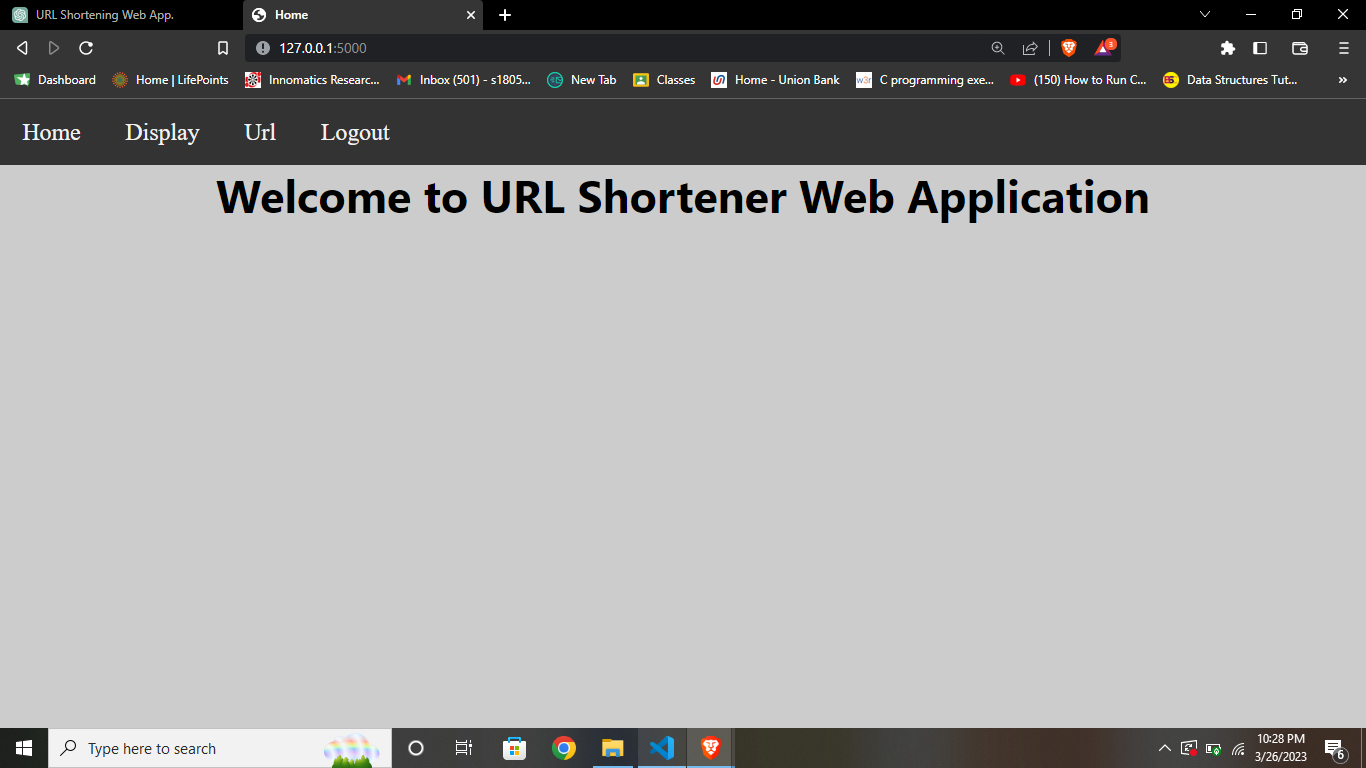
flask db upgrade

Step-10

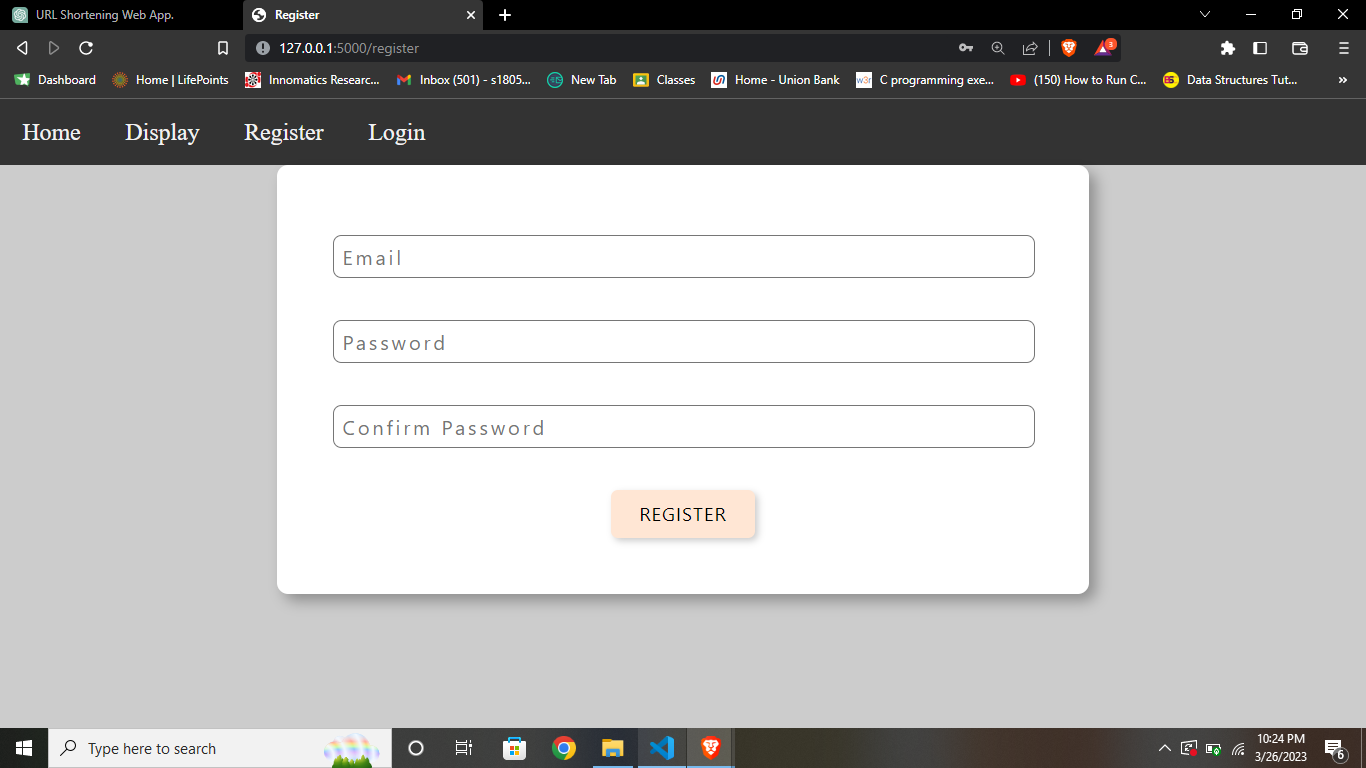
Run the Python code app

python app.py

Step-11

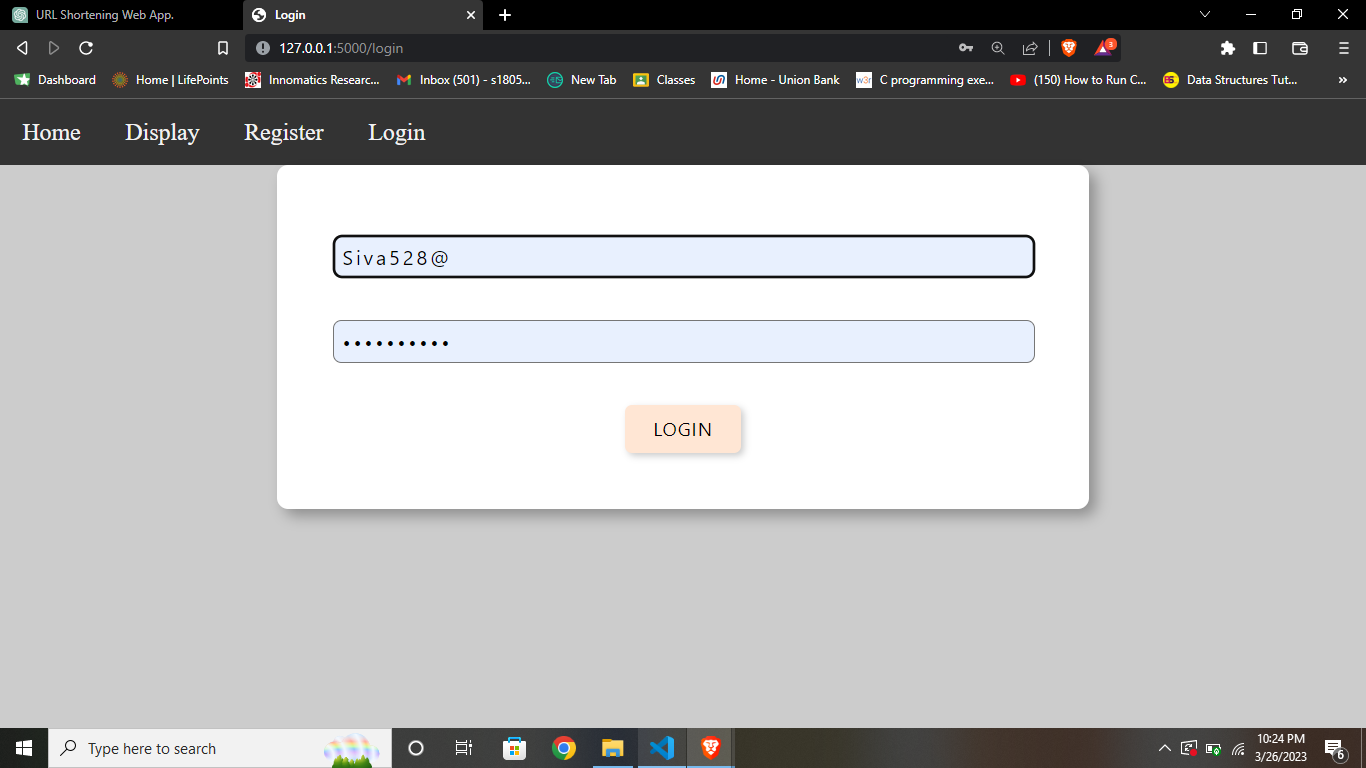


Register



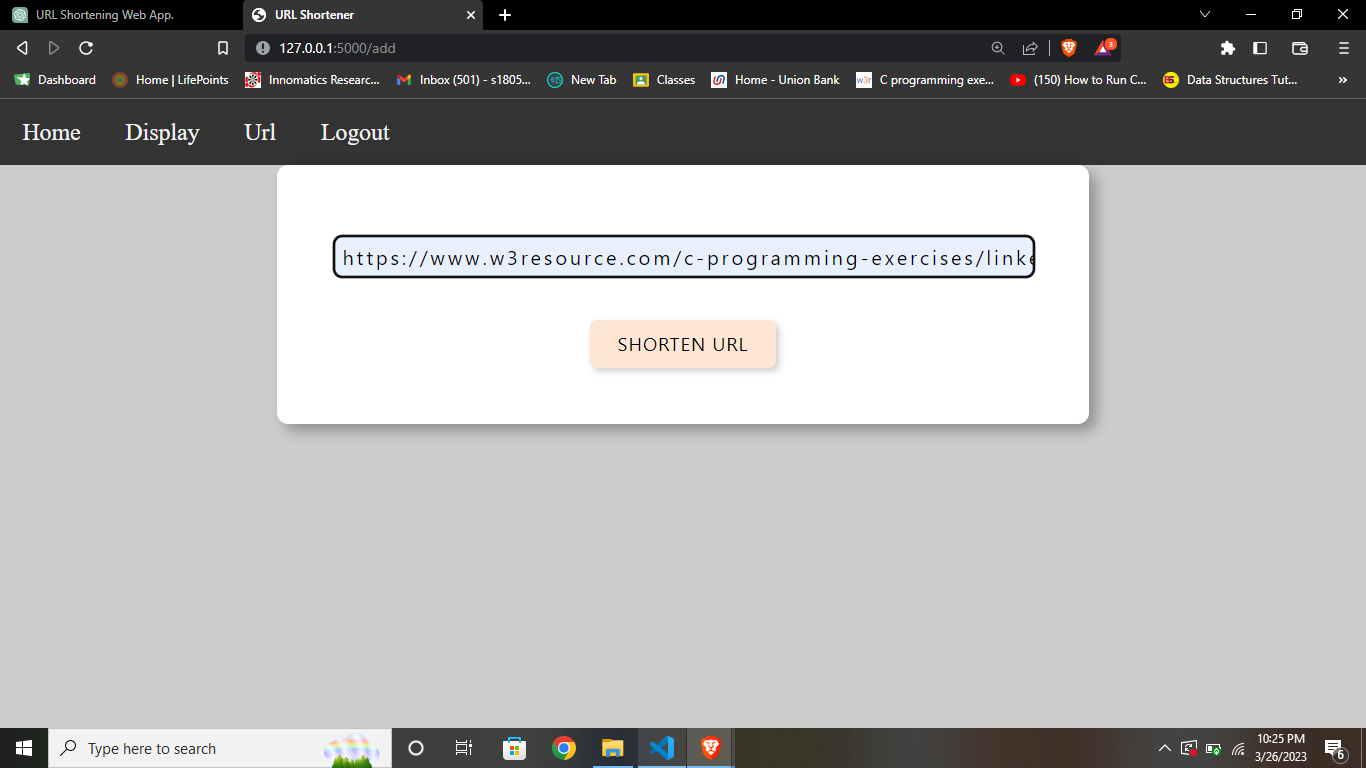
Step -12

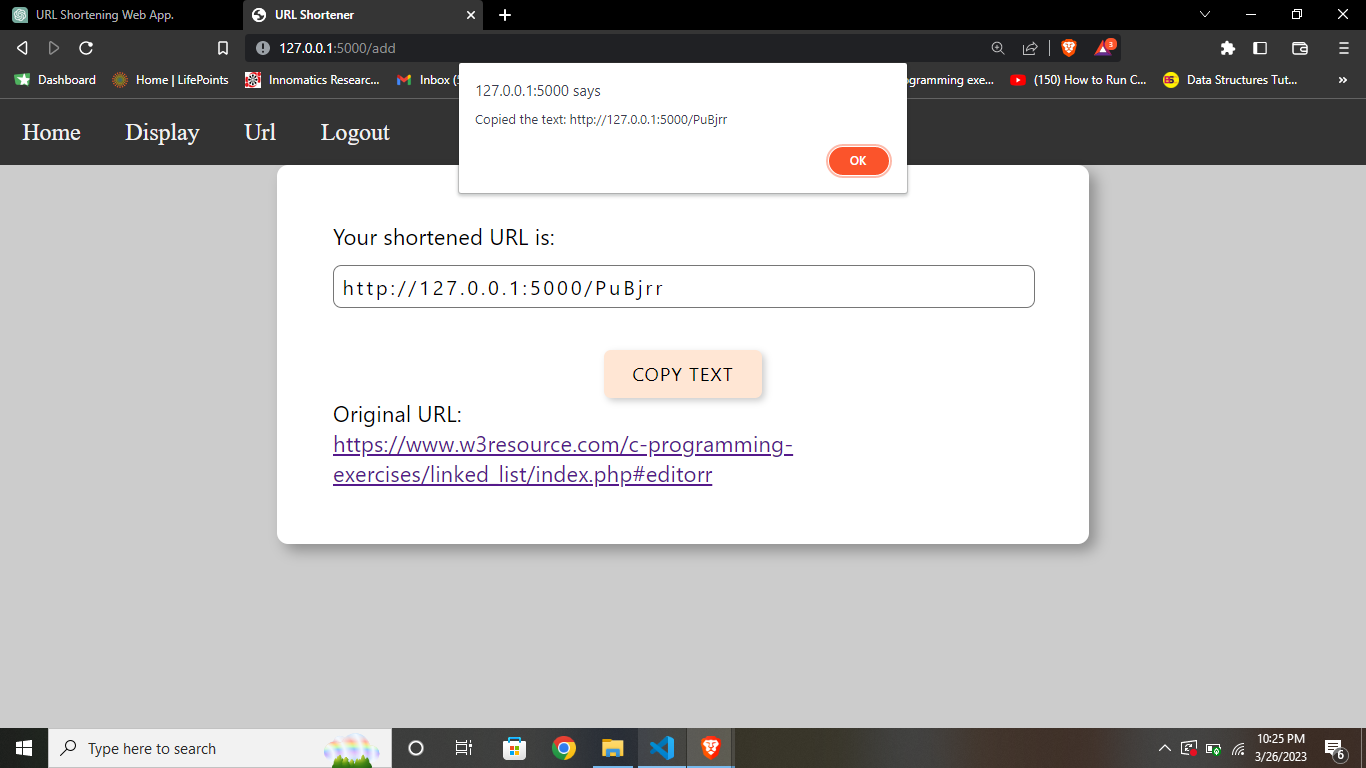
Login



Url Option Enabled Now

Shorten Url





History

