URL Shortener App Report - By Srushti Sonavanae

A. What will our Web app do(Objectives)?

As the name suggests, it shortens URLs.

Users can also save URLS by coming to the web app.

B. Why do we need a URL Shortener?

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

C. The project consists of the following parts:

Frontend (done with HTML, CSS, JavaScript, and Bootstrap)

Backend -Flask (Python)

Backend -Database ORM (SQLAlchemy)

Backend -Database (SQLite)

D.Front-EndInformation:

-> The front-end consists of 2 web pages:

Home Page -A page will be shown where the user can enter the URL he/she wants to shorten. After the ‘shorten’ button is clicked, the shortened URL is displayed in the text field which the user can copy using the copy button.

History Page -Containing all the Original URLs along with the Shortened URLs.

E.Project Workflow

Users can enter the URL they want 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.

After the ‘Shorten’ button is clicked, the entered URL 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/she entered in our web app with their shortened URL.

The app also verifies whether the URL entered by the user is correct.

F.Approach

To build the app following packages were used: i.)flask ii.)flask_sqlalchemy iii.)flask_migrate iv.)os v.)validators vi.)random, string

Steps:

1. Create a simple app.py file and import the necessary modules from the provided packages, as well as code the basic flask template.
2. Make a folder called "templates" inside the project folder, which is the root directory of app.py.
3. Inside the templates folder, create three HTML files with the names "layout.html," "home.html," and "history.html."
4. Navigate to the root directory of app.py and create a new SQLite database with admin access using the command prompt, with three fields named "id" (Primary Key), "full_url," and "shorten_url."
5. "layout.html" should contain all of the fundamental template code, including the navigation bar, as well as inheritance blocks.
6. Create "home.html" and "history.html" using the inheritance concept from "layout.html," and include relevant requests, forms, and display content.
7. Add CSS styling to both HTML files.
8. Add a button in "history.html" to copy the shortened URLs using JavaScript code snippets.
9. In app.py, first use the "os" module to locate the app.py's root directory (base folder) and save it in a variable.
10. Configure the SQLAlchemy ORM configurations and utilize the variable used to store the path using the os module to provide the SQLite database path.
11. Create a class URL and a table with the relevant columns as described in step 4.
12. Create three routes "/", "/history", and "/finalurl>" with the functions home(), history(), and redirection().
13. "home()" -takes an input URL in the form of a POST request and validates it using the validators module; if the URL is invalid, an error is displayed; otherwise, it checks whether the entered URL is available in the database. If it is present, it returns its previously constructed shortened URL; otherwise, it generates a random string of 5

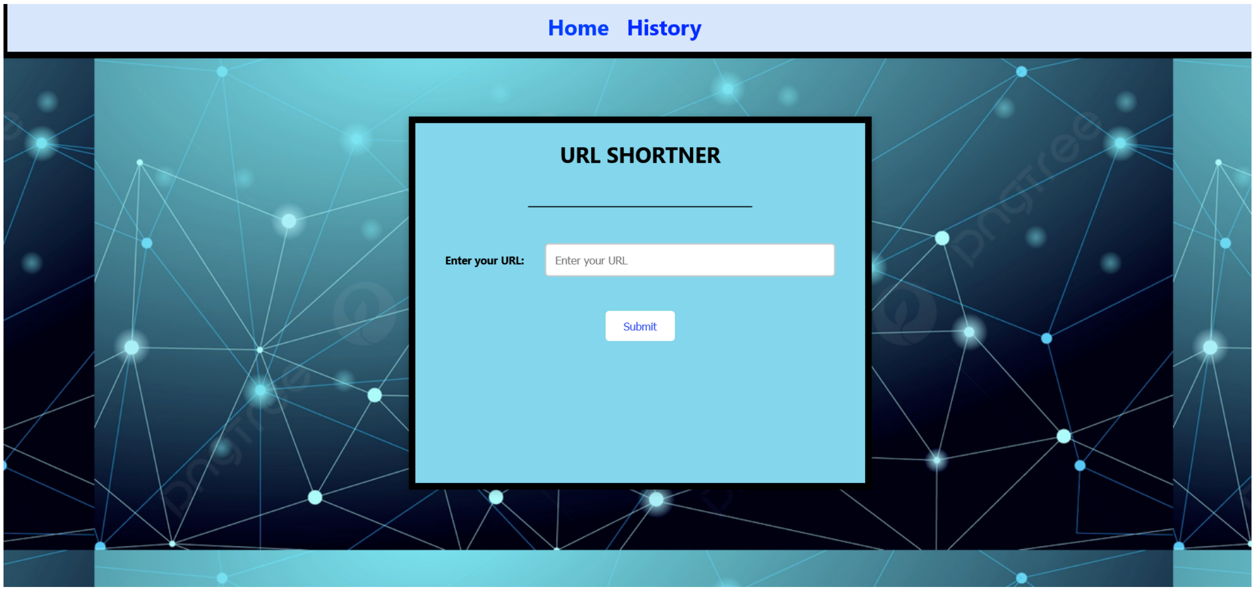
characters and attaches it to the basic "127.0.0.1:5000" URL (if it is not otherwise) and keeps this link in a variable.

If the link in a variable is already present in the database, it produces new URLs until a unique URL is identified, attaches the Original URL and Shorten URL together, and adds it as an object to the database. It then displays the Shorten URL beneath the Original URL, which may be copied using the Copy URL button.

14. "history()" - this retrieves all database rows and displays each original URL and Shorten URL ever created row-by-row in a table in Frontend.

15. "redirection()" -this function is used to give the feature of retrieving the original URL from the database when a short URL is entered and opening the page associated with that Shorten URL.

Screenshot:



GitHub Link:

https://github.com/Srushti-S/Innomatics_Research_Labs_Internship/tree/main/Task_5