URL_SHORTNER REPORT

- D. Manisha Kranthi

The objective of the project is to create a web app which takes a URL as an input and shortens it. Also stores the values entered by the user in the database.

What will our Web app do (Objectives)?
1. As the name suggests, it shortens URLs.
2. Users can also save URLS by coming to the web app.

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.

The project consists of 2 parts:
1. Front end (done with HTML, CSS, and Bootstrap)
2. Back end - Flask (Python)
3. Back end - Database ORM

Front-End Information
The front end consists of 2 web pages:
1. 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.
2. History Page - Containing all the Original URLs along with the Shortened URLs.

Project Work flow
1. 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.
2. 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.
3. Try to verify whether the URL entered by the user is correct or not.

STEPS FOLLOWED TO CREATE URL SHORTNER WEB APP:

1. Built as basic web app in app.py format.
2. Imported the modules required such as flask, flask_alchemy, flask_migrate, validators, OS, random and string.
3. Created a folder inside the project folder with the name ‘templates’.
4. Created a basic html files inside the templates folder called home.html, history.html and layout.html.
5. Layout.html contains all the basic template codes and the layout.
6. Created home.html and history.html using the inheritance from the layout.html. It has the relevant requests, forms and display content.
7. Added a button in history.html to copy the shortened urls.
8. In the root directory of app.py created a new database using command prompt with 3 fields such as “id”, “full_url”,”shorten_url”.

9. Configured the SQLAlchemy ORM configurations and used the variable which was used to store the path using os module and gave path of the database.
10. Built a class url and created a table with relevant columns.
11. Created 3 routes with functions home(),history() and redirection().
12. ‘home()’ takes the url in the form of post and returns the shorten url.
13. ‘history()’ displays the original url and shorten url in the table by fetching the history from the database.
14. ‘redirection()’ opens the relevant page associated with the shortened url after shortening the original url entered by the user.