URL Shortener App Report

---

1. 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.

2. Why do we need 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.

3. The project consists of following parts:

- Frontend (done with HTML, CSS and Bootstrap)
- Backend - Flask (Python)
- Backend - Database ORM (SQLAlchemy)
- Backend - Database (SQLite)

4. Front-End Information

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.

5. 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 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.
- Try to verify the URL entered by the user is correct or not.

6. Modules imported

During the development of the project following modules were imported-

- os
- Flask, request, render_template, redirect, url_for
- flask_sqlalchemy, SQLAlchemy
- flask_migrate, Migrate
- wtforms, validators
- string
- random, choice

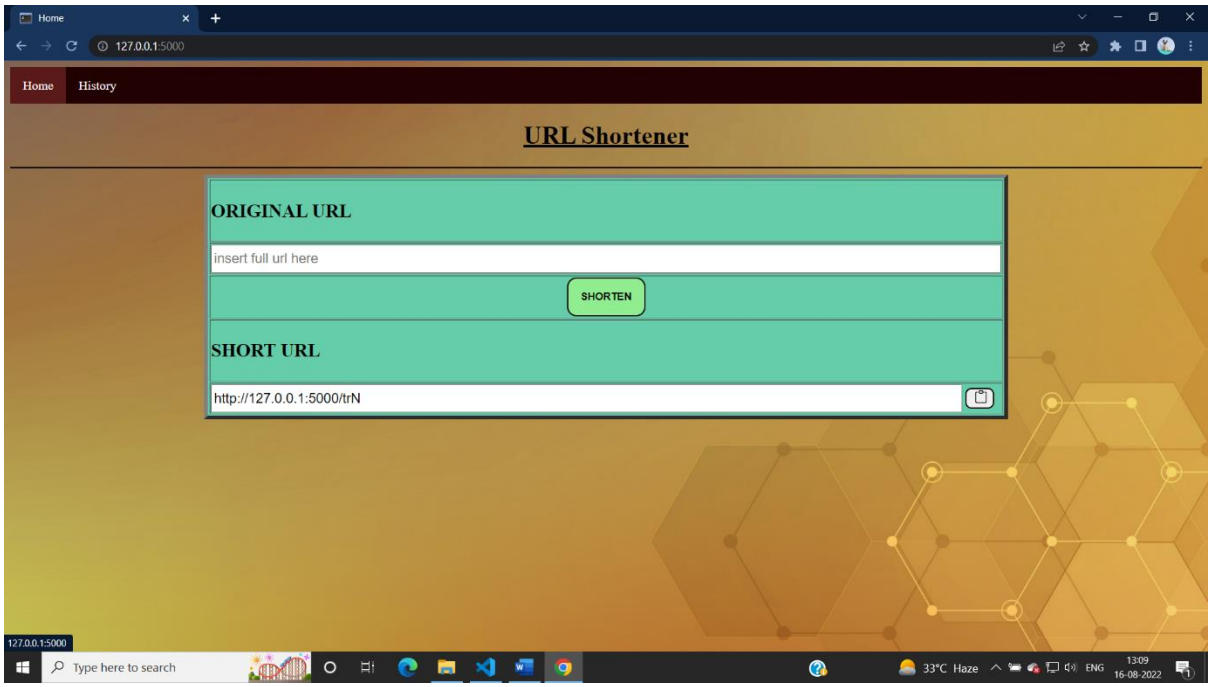
7. Steps to be followed for implementation of application (Approach)

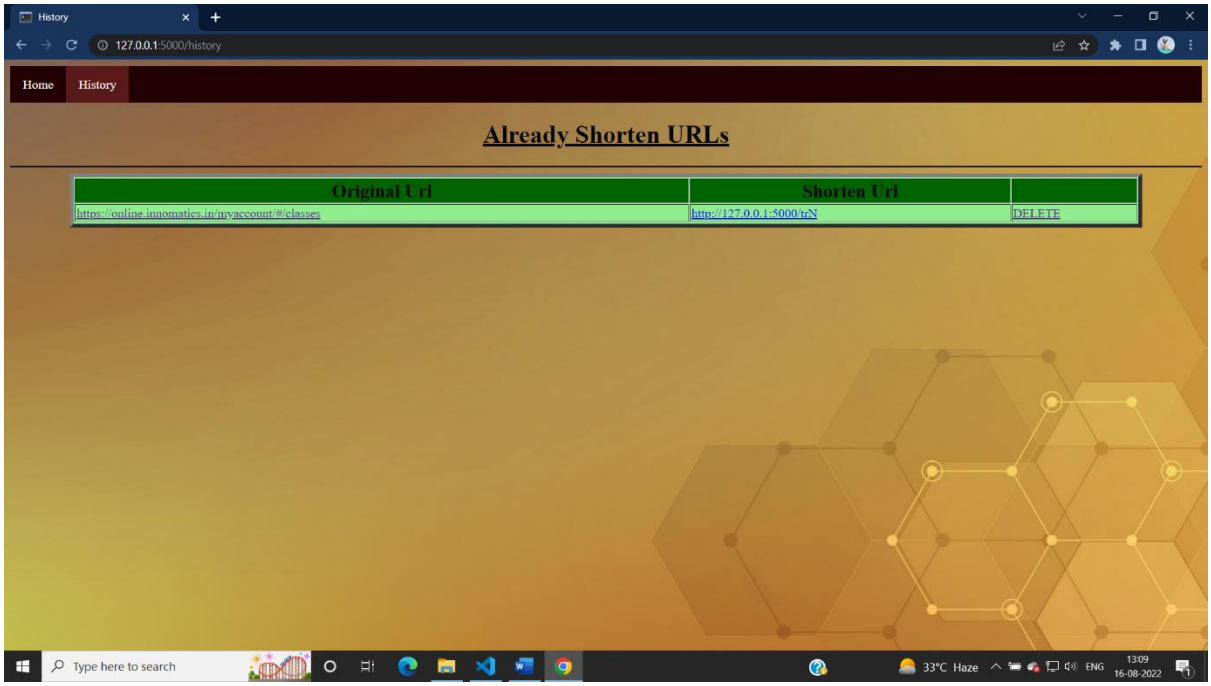
- Create a folder named with the Application name (here, **URL Shortener**).
- Build a basic python file (**app.py**) and import all the required modules as mentioned above from their packages then code the basic flask template.
- Create a folder named “**templates**” under the main folder directory (i.e., under **URL Shortener**).
- Under templates folder create required “.html” files
  Here,

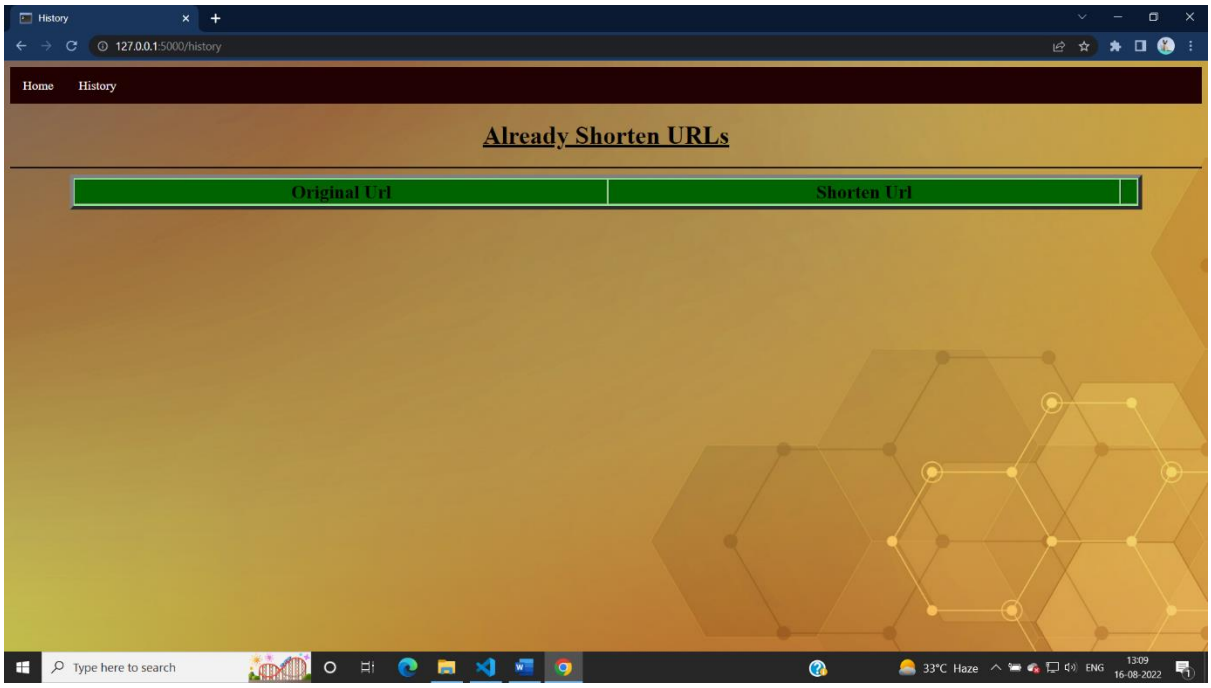
    - **layout.html** – It contains the overall common designing and styling codes of all the web pages inside this application.
    - **home.html** – This page inherits layout.html and then put its own functionalities in it. It contains a form which accepts a long URL and display the short URL of the given long URL. It has a copy button to copy the shorten displayed URL.
    - **history.html** – This page also inherits layout.html and then puts its own functionalities in it. In this page, you can see all the already shortened URLs and delete those which are not in use anymore.
- Give CSS styling to all the html files.
- Now, go to app.py and store it ina variable using os module.
- Establish connection between database and backend using SQLAlchemy through configuration of SQLAlchemy ORM and use the variable to give the path to SQLite database.
- Pass the entire application into SQLAlchemy and store it in a variable further used for database.
- Create a model class (here, **Url**).
- Provide a tablename and columns/fields to the table (here, id, long_url, short_url).
- To add database SQLite file in main folder, we have to run 3 commands through command prompt as administrator to access to the 3 fields.
- Create required routes (here, 4, i.e., “/”, “/history”, “/<finalurl>”, “/delete/<int:id>”) with their binding functions home_page(), history_page(), redirection() and delete().Below details of functions-
    - **home_page()** – this function receives the value of URL from form in home.html page as form for POST method and check whether the provided URL is valid or not using validators module, if not then display an error message otherwise shorten the URL if also the provided URL is not already shortened and not present in database (i.e., history). If URL already present in history then return the same shorten URL. Finally, the URL display in a non-writable textbox and you can now copy the shorten link using copy button beside the textbox.

- **history_page()** – this function fetches all the details from database and show all the entered original URLs with their shorten URLs in the history page and also you can delete a URL if you don't want to use it anymore.
- **redirection()** – this function is used for redirecting to the relevant site when click on short URL if URL is valid.
- **delete()** – this function is used to delete a row from the history page by fetching its id from the database and remove all the details related to the respective id.

8. Screenshots of running application







Github link: https://github.com/dhreeti414/URL_Shortener

LinkedIn link: https://www.linkedin.com/in/dhreetikesharwani/