URL Shortener Report

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. Frontend (done with HTML, CSS, and Bootstrap)
2. Backend - Flask (Python)
3. Backend - Database ORM

Technical details

The project was developed using the Flask web framework and SQLite3 database. The project files include the following files:

- `app.py`: The main file that contains the Flask application code.
- `templates/index.html`: The HTML template for the home page of the application.
- `templated/display.html`: The HTML template for the display page of the application.
- `templated/header.html`: The HTML template for the `<head> ... </head>` of the other HTML documents..
- `templated/navbar.html`: The HTML template navbar of all HTML pages.
- `templated/table-row.html`: The HTML template for the table rows on the display page for each URL entry.
- `static/styles.css`: The CSS file for the application.

The Flask application code includes the following routes:

- `"/"`: The home page of the application that displays a form for the user to enter a URL to shorten. The shortened link is automatically copied to the user's clipboard.
- `"/display"`: The route that displays all URLs entered by the user.
- `"/<short_url>"`: The route that redirects to the original URL corresponding to the short URL.

The SQLite3 database is used to store the original URL and the corresponding short URL. The database has a single table with the following columns:

- `id`: A unique integer ID for each URL.
- `original_url`: The original URL submitted by the user.
- `short_url`: The short URL generated by the application.

Project Workflow

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. (Do some googling to find out how to make it possible)

Conclusion

The URL Shortener project was successfully developed using Python, Flask, and SQLite3. The application allows users to shorten any valid URL and redirects them to the original URL using the short URL. The project code and database files are available on GitHub. Further development of the project can include additional features such as user authentication and analytics tracking.

The URL Shortener project code and database files are available on GitHub.