Basic URL Shortener Web Application using Flask and SQLite

Name – Priyanshu Semwal

ID – IN125036111

Introduction

In today's digital world, sharing links has become a routine task. Many websites generate long URLs which are inconvenient to share via email, messaging apps, or social media. This is where URL shorteners become extremely useful. A URL shortener is a tool that converts a long URL into a shorter, more manageable version.

Objectives of the Project:

1. To shorten long URLs for easy sharing.
2. To save and manage previously shortened URLs for reference.
3. To create a user-friendly web application using modern web technologies.

Real-World Examples:
Some popular URL shortening services include:

- **Bit.ly**
- **TinyURL.com**
- **Rebrandly**

Frontend

- **HTML (HyperText Markup Language)**
  Used to structure the content of web pages, including input forms for URLs, buttons, and links to history pages.
- **CSS (Cascading Style Sheets)**
  Provides styling for the web pages, such as colors, fonts, spacing, and layout.
  Example: making the home page look clean and readable.
- **Bootstrap**
  A popular CSS framework that helps create **responsive and professional-looking web pages** quickly.
  In this project, Bootstrap is used to:
  - Style the form inputs and buttons
  - Make the table on the history page look clean
  - Ensure the pages work well on different screen sizes

Backend

- **Python**
  A versatile programming language used for the backend logic.
  Handles URL shortening, validation, and redirecting.
- **Flask**
  A lightweight Python web framework used to build web applications.
  It allows creating routes (pages) such as / (Home) and /history (History page), and connecting them with the backend logic.

Database

- **SQLite**
  A lightweight, file-based relational database used to store:
  - Original URLs
  - Shortened URLs
- **SQLAlchemy (ORM)**
  An Object Relational Mapper (ORM) for Python, which allows **interacting with the database using Python objects instead of SQL queries**.
  Benefits include:
  - Easy database management
  - Less chance of SQL errors
  - Cleaner code

Other Technologies

- **JavaScript**
  Used to implement the **copy-to-clipboard feature**, allowing users to click a button to copy the shortened URL instantly.
- **Browser / Chrome / Edge**
  Used to run and test the web application.

Features Implemented

1. **Home Page**
    - Input field to enter the URL to be shortened.
    - **URL Validation:** Ensures that the entered URL is in the correct format (must contain a . and proper structure).
    - **Shortened URL Generation:** Generates a unique 6-character short code for the URL.
    - **Copy-to-Clipboard Button:** Allows users to copy the shortened URL with a single click.
    - **Error Handling:** If an invalid URL is entered, an error message is displayed.
2. **History Page**
    - Displays a table of all previously shortened URLs.
    - **Original URL** and **Short URL** columns are shown.
    - Clicking on a short URL redirects the user to the original URL.
    - Navigation link back to the Home page for convenience.

submission/

├── app.py # Main Flask application

├── models.py # Database models (URL table)

├── urls.db # SQLite database storing URLs

├── templates/

│ ├── home.html # Home page template

│ └── history.html# History page template

User opens Home Page

|

V

Enters URL → Clicks Shorten

|

V

Validation (Check format)

|

V

Short code generated → Saved in Database

|

V

Shortened URL displayed

|

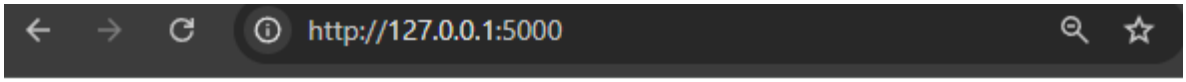
V

User can copy URL OR visit /history

|

V

History Page shows all URLs



Basic URL Shortener

Enter URL Shorten

View History

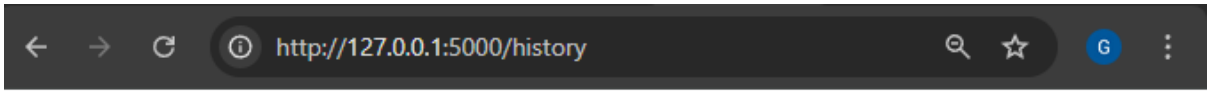
Basic URL Shortener

google.com Shorten

Short URL:

http://127.0.0.1:5000/P85F74 Copy

View History



URL History

Original URL	Short URL
https://Google.com	http://127.0.0.1:5000/sTHBwE

Go Back