INNOMATICS
RESEARCH LABS

Internship Project Report

July 2022(Batch)

Title:

Web Application for URL Shorten

Submitted By:
Vicky Raghuwanshi

Under the Guidance of:
Kanav Bansal
(Chief Data Scientist)
Innomatics Research Labs, Hyderabad

GitHub | LinkedIn

- **Executive Summary**:

  URL shortening is a technique on the World Wide Web in which a Uniform Resource Locator (URL) may be made substantially shorter and still direct to the required page. This is achieved by using a redirect which links to the web page that has a long URL. For example, the URL "https://www.google.com/search?q=flask+sqlalchemy&rlz=1C1RXQR_enIN968IN968&oq=flas&aqs=chrome.0.69i59j69i57j0i271j69i60l5.1773j0j1&sourceid=chrome&ie=UTF-8" can be shortened to "https://example.com/Hexacode ", and the URL "https://en.wikipedia.org/wiki/gjhdvi4Jfbfdknhjfbfgr57HYgugdf " can be shortened to "https://w.wiki/2u7". Often the redirect domain name is shorterthan the original one. A friendly URL may be desired for messaging technologies that limit the number of characters in a message (for example SMS), for reducing the amount of typing required if the reader is copying a URL from a print source, for making it easier for a person.

- **Objective:**

  The Objective of the project is to build the system which allows user to enter a URL, which he wanted to shorten. The URL he provided should be a valid URL unless the application would not shorten the URL he provided. If the URL he provided is valid or with a status code of 200, he will receive a shorten URL which he can use to redirect to the original page. The original URL plus the shorten URL will be saved into database. If the user enters the URL which he has shortened before then he will receive the short URL from the database. The shorten URL will redirect the user to the original URL page

- **Introduction**:

  This project based on Redirecting original URL to tiny URL by using this web application of backend web application server done by using Python programming with Flask Module and many more modules also this project contains flask_sqlalchemy module to their backend database. There are several reasons to use URL shortening.

- **Techniques:**

  In URL shortening, every long URL is associated with a unique key, which is the part after its top-level domain name. For example, `https://tinyurl.com/m3q2xt` has a key of `m3q2xt`. Not all redirection is treated equally; the redirection instruction sent to

a browser can contain in its header the HTTP status 301 (Moved Permanently), 302 (Found), 307 (Temporary Redirect) or 308 (Permanent Redirect).

There are several techniques to implement a URL shortening. Keys can be generated in base 36, assuming 26 letters and 10 numbers. In this case, each character in the sequence will be 0, 1, 2, ..., 9, a, b, c, ..., y, z. Alternatively, if uppercase and lowercase letters are differentiated, then each character can represent a single digit within a number of bases 62 (26 + 26 + 10). In order to form the key, a hash function can be made, or a random number generated so that key sequence is not predictable. Or users may propose their own custom keys. For example, `https://example.com/product?ref=01652&type=shirt` can be shortened to `https://tinyurl.com/exampleshirt`. Not all URI schemes are capable of being shortened as of 2011, although URI schemes such as `http`, `https`, `ftp`, `ftps`, `mailto`, `mms`, `rtmp`, `rtmpt`, `ed2k`, `pop`, `imap`, `nntp`, `news`, `ldap`, `gopher`, `dict` and `dns` are being addressed by such services as URL shorteners.
Typically, `data:` and `javascript:` URLs are not supported for security reasons (to combat attacks like cross-site scripting and session hijacking). Some URL shortening services support the forwarding of `mailto` URLs, as an alternative to address munging, to avoid unwanted harvest by web crawlers or bots. This may sometimes be done using short, CAPTCHA-protected URLs, but this is not common

- **Python Programming language:**

  Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

  Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library

- **Flask (Web Framework):**

  Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

- **HTML5:**

  HTML5 is a markup language used for structuring and presenting content on the World Wide Web. It is the fifth and last major HTML version that is a World Wide Web Consortium (W3C) recommendation. The current specification is known as the HTML Living Standard. It is maintained by the Web Hypertext Application Technology Working Group (WHATWG), a consortium of the major browser vendors (Apple, Google, Mozilla, and Microsoft).

- **Cascading Style Sheets (CSS):**

  Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

  CSS is designed to enable the separation of presentation and content, including layout, colours, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .CSS file which reduces complexity and repetition in the structural content as well as enabling the .CSS file to be cached to improve the page load speed between the pages that share the file and its formatting.

  Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.

- ## Containing Packages(modules):

  from flask import Flask, render_template, request, redirect

  import pyperclip as pc

  from flask_sqlalchemy import SQLAlchemy

  from flask_migrate import Migrate

  import os

  import string

  import random

- ## Working Phases

  The application working is divided into four phases:

  1. **URL input**: The goal of the URL input is to accept the valid URL from the user which he wants to shorten.
  2. **URL shortening**: The goal of this phase is to map the given URL to 6 alphanumeric characters.
  3. **URL Mapping**: The goal of this phase is to map the original URL and its short URL into the database, so the same short URL for the original URL can be used in future.
  4. **URL redirection**: The goal of this phase is to enable shorten URL to redirect to the original URL web page.

- ## Feature Scope

  I can add two more features to make this application more interactive:

  1. **Registration feature for the new user (Signup)**: If a new user wants to use this application, he has to register himself/herself with required credentials to createhis account.
  2. **Login feature for the old user (Login)**: old user can use this application by just loggingin into his account.
  3. **Adding Buttons:** In History page, I am going to add the **Delete** and **Copy** button in the URL's Table in the History Page.