Web Services and Cloud Based Systems - Assignment 1

Arumoy Shome
VU ID: 2636393

Shruti Rao
VU ID: 2631481

Group 13

Assignment 1.2

In a URL shortening service, an encoder and a decoder maps long URL Strings to short Strings . Further, a web framework redirects the short URL to the original URL [1]. *Flask* - the Python micro-framework was used to set up a RESTful url-shortner service [7]. The string encoding and decoding is handled by *short_url* - a pre-existing Python implementation [4]. It was decided to use *short_url* since it uses a **bit shuffling approach** which is more comprehensive and prevents predictable generation of URLs [4]. The application also uses a *SQLite* database to store URLs. *SQLite* was considered a natural database choice as it is built into Python3 and convenient for small applications [7].

Database

`schema.sql` is the database schema that creates an empty `urls` table to store and retrieve URLs.

`db.py` creates a connection to the database when handling a request. This connection is closed before the response is sent. All queries and operations are performed when the connection is created [7]. Python functions additionally run **SQL** commands from `schema.sql` to the `db.py`.

Blueprint

Requests to the application are handled by a *view* function [7]. The data returned from the view is the response [7]. `shortner.py` is defined as a blueprint that organizes related views. `shortner.py` thus gets registered with the application, defines all services, endpoints and the manner in which requests are handled.

Updates sine demo

During the demo we had an incomplete implementation of the service, these have now been implemented. Please follow the instructions outlined in the README for testing the service. Please note that a limitation of the current implementation is that the service lacks any error handling for failed database transactions. So for instance, if we try to make a `POST` request to / with an already existing url, the service will fail.

Extending Application for Multiple Users

To extend the URL shortening application to support multiple users, the database needs to have an additional `users` table. Here, in the `users` table, a **unique key** can be added for every unique user.

Further, the pre-existing `urls` table can be extended to have a **creater ID** that would be equivalent to the unique user ID. The web page would then render the URLs for a specific user.

References

[1] cawcaw et al. *How do I create a URL shortener?* URL: https://stackoverflow.com/questions/742013/how-do-i-create-a-url-shortener.

[2] *Flask-Spyne*. URL: https://pypi.org/project/Flask-Spyne/.

[3] *Python SOAP client*. URL: https://python-zeep.readthedocs.io/en/master/.

[4] *short$_u$rl*. URL: https://pypi.org/project/short_url/.

[5] Tutorialspoint.com. *WSDL Example*. URL: https://www.tutorialspoint.com/wsdl/wsdl_example.htm.

[6] Vpulim. *vpulim/node-soap*. Mar. 2019. URL: https://github.com/vpulim/node-soap.

[7] *Welcome*. URL: http://flask.pocoo.org/.