Assignment 1.2

RESTful webservice, url-shortener

April 11, 2019

Students:
Kiran Tjikhoeri
10800158

Berk Aydasgil
12492930

Talha Shaikh
12503819

1 Design and implementation

For this assignment we had to implement a URL shortener using the GET, POST, PUT and DELETE methods.

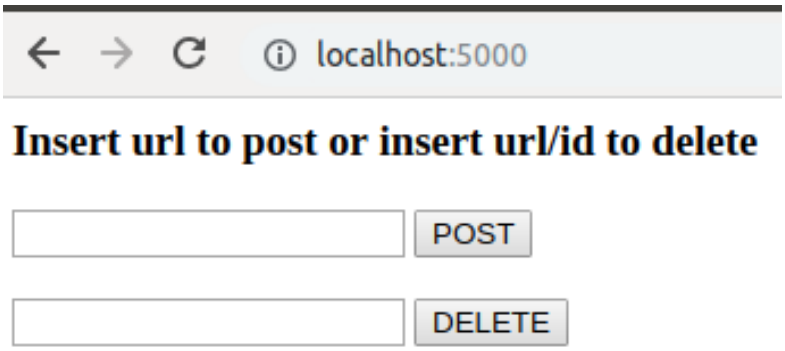

Figure 1: screenshot of the startpage

It works as follows:

- When the user goes to the start page it will view two buttons with which he can post a new url to get a new id or delete an existing url. The startpage:
  localhost:5000

- When the user wants to see all the current URLs with the corresponding id's, it can be seen using the following url:
  localhost:5000/all
- When the user wants to find a URL by its id (GET), the user needs to fill in the following url:
  localhost:5000/<id>
- When the user wants to find the id of a URL (GET), the user needs to fill in the following url:
  localhost:5000/<url>
- When the user wants to insert a new URL creating a new id, the user needs to (POST) the new url using the input bar on the start page:
  localhost:5000
- When the user wants to DELETE a URL given a URL or id, it needs to use the corresponding input bar on the startpage.
  localhost:5000
  But since HTML only accepts POST and GET, this delete is not done using the DELETE method. However, we implemented a function which can be used as follows:
  An element can be deleted with the id:

```
curl -i -H "Content-Type: application/json" -X DELETE -d
'{"id": "<insert existing id to delete>"}' http://localhost:5000
```

  or with the url:

```
curl -i -H "Content-Type: application/json" -X DELETE -d
'{"url": "<insert existing url to delete>"}' http://localhost:5000
```

- When the user wants to PUT a new value for an existing id, it can be done as follows:

```
curl -i -H "Content-Type: application/json" -X PUT -d
'{"url": "<new url>", "id":"<existing id>"}' http://localhost:5000
```

We used Python 2.7 and the framework Flask. We also used json instead of XML as the content type. Since HTML only accepts GET and POST methods, we did the DELETE and PUT method without a button in HTML.

However we did implement a delete button using the POST method.
The unique id's the program creates is just an int starting from zero and this increases with one every time a new url is posted, this value is then encoded to a base62 value.

2 Answer to question 3

How would you implement a URL-shortener for multiple users?
When multiple users use such a service, it is important to take into consideration how many people will use it and how many available unique id's you have. If you have more people using it than the amount of unique id's possible then it should be a good idea to keep these shortened links active for a specified amount of time. So when this time is exceeded, the unique id can be used or modified for another url. Another important thing is to use a real database. In this assignment we just used a dictionary and that is not that useful when dealing with lots of urls.