An RESTful URL-shortener service based on Flask

Group 16: You Hu 12036234, Wenchen Lai 12482625

URL shortening is the technique in which a URL may be made substantially shorter and still direct to the required page. This service provides a mapping from self-defined URL or identificator to target URL, and provides the interface for updating, deleting and adding mappings.

The framework we choose is Flask on which a light service can be established. The mappings are stored in python builtin dict structure, and all operations are based on it.And the mappings will be cleaned after a restart, service has no state The returned messages are structured in *response*, *redirect(301)* and *abort(404)*.

According to the requirements, the service should provide the required functions with two forms of the path: root path and path with id. In RESTful, the resources are bound to paths, therefore we implement those functions as the methods of resource objects and bind them to different paths.

For the root path, the service provides three methods: GET, POST and DELETE. By invoking them, the functions listed below are achieved.

- GET: return the keys(ids) and code 200.
- POST: the method fetches the argument which is URL; if the received URL is not valid in format(tested with regular expression), return “error” and code 400; otherwise create a new mapping from id equal to maximum id plus one to URL, return the id assigned to URL and code 201.
- DELETE: clean all the mappings and return code 204.

For the path with a particular id, the service provides three methods: GET, PUT and DELETE. By invoking them, the functions listed below are achieved and passing id as an argument for the function. If the id requested is not in the dict, the function will abort in advance with code 404.

- GET: return the URL mapped by the id and redirect to the page by URL with code 301(return *redirect* object).
- PUT: the method fetches the argument which is URL; if the received URL is not valid in format(tested with regular expression), return “error” and code 400; otherwise return code 200 and update the mapping with new URL.
- DELETE: delete the mapping by the id.

Above is how we implement the service with Flask-restful.

When considering implement URL-shortener for multiple users, our implementation should be changed. First, we should change our architecture to store mappings in the database instead of dict in memory to maintain the states, according to the US patent No. US 2015/0161282 A1. For performance, considering that python does not provide “real” multi-thread functionality because of the GIL, we should open multiple threads for each thread there is one service is bound to a unique port and use DNS service to let users access to different ports to get service with the same URL or path. Of course, there is also a place for authorization authentication, which means only particular users can access some certain addresses.