INFO253: Web Architecture (Technical Report)

Kevin Kao
kkao@berkeley.edu

For the INFO253 course project, I designed a URL shortening website called scribble.ly, comparable to services like bit.ly and goo.gl. The initial project design was very minimal, with limited features for a URL shortening website. The initial features only allowed for alpha characters and only provided validation for that. The service also only allowed for manual inputs of short codes by the user.

To improve upon the initial specs of the project, I implemented a few additional, handy features. The first feature I added was support for both alphanumeric characters, case-sensitive. This allows for a larger number of URLs to be shortened, since we can include combinations of numbers into the six-length short code. The feature mostly involved adding extra allowances on the server side (Flask). With the addition of this, I also added in additional validation on short code generation on the client-side, which would ensure that all user-inputted short codes are alphanumeric instead of only alpha characters. This mostly involved additional jQuery only. In addition to this, I added automatic short code generation. Most URL shortening services automatically generate URLs for the user so that there are fewer collisions when a user tries to manually input his/her own. To automatically generate URLs, on the server side, I used the built-in random string generation method in the string library of Python and randomly generated 6 characters as a string.

An optimization I implemented into scribble.ly was a “fuzzy” join on URL short codes. URLs such as “www.google.com”, “http://www.google.com”, and “google.com” all point to the same location, so to optimize space and reduce redundancy, I implemented the fuzzy join of URLs that looked for similar URLs such as these and aligned them all with the same short code. A con of this feature is that users cannot generate unique short codes for the same long URL, but since there is no user-based system implemented into scribble.ly, the impact is small. A good feature to consider in the future would be an option to opt-out of this collapsing of URLs, perhaps for the purpose of tracking different short codes on different audiences.

The final feature was real-time tracking analytics for short codes. The analytics was a simple one that only kept track of when a short code was visited. This information was stored in a new database on disk and could be queried. Aggregations were performed on a per day basis for each short code. This feature is useful for users who want to have a better understanding of which URLs their audience is most interested in,

or which days of the week do people visit the URL more, etc. This basic tracking provides enough information to do that, although more complicated tuning features could be implemented in the future (e.g. “zooming” in and out for the scale from day to hour or day to week).

To present the analytics to the user client, I utilized D3.js, a JavaScript graphics and animations library used to draw fancy graphs. Specifically, I used a wrapper around D3 called Rickshaw.js. Rickshaw.js only provides an easy way to initialize a line graph; I tackled a larger problem of updating the graph. In order to fetch the latest analytics information for a user who entered a short code on the analytics page, I added jQuery code that made GET requests at continuous intervals of a few seconds to query for the latest information. The line graph would then be updated with the new information, as long as the user stays on the page. The graph was scaled to show approximately 7 days of information, starting from the current day and going backwards. Smooth transitions from D3.js were used.

A big design aspect of my project was to follow the principles of building a single page application (SPA). Unfortunately, due to the mysterious positioning of SVG objects on an HTML page, I was unable to completely do so and had to place the analytics tracking feature on its own separate page. In the future, with more experience with CSS, I hope to transform this back into one page.