Rahul Rajagopalan
6.170

Link Shortener Testing

The application was tested primarily through the Flask/unittest approach. This is because most of the application logic is on the server side, so testing is more straightforward when also run in Python. Also, by not testing from the client side, the application avoids the issue where changes to the visual design cause tests to break; this is undesirable when the tests' intent is to verify the controller and model.

The view was mostly tested manually because it is tricky to programmatically evaluate a user interface. In the case of this application, the JavaScript was entirely presentational logic (leaving most computation to the server), so it was tested manually. I created multiple users and maintained sets of shortened URLs, much like end-users would do, to see if any issues occurred.

The automated tests cover every entry point of the application, and include registering/logging in users, shortening URLs with both inputted and generated aliases, viewing shortened URLs, visiting shortened URLs, and looking up analytics. This inspects every use case of the application, so I consider them to be sufficient to verify the application.