To create this website, we needed to create a registration, log in, and a spell check page. I also created a home page that allowed for navigation between these pages. The home page also gave the ability to log off if the user is logged in.

Because we are giving the user certain abilities once they log in, we are using sessions instead of cookies to hold onto this data. This is because it is more secure for the user information to be stored in sessions where it is harder to read than if they were stored in cookies. So sessions are created when the user logs in and the session exits when the user logs off.

CSRF is an attack where the you can use the victim’s credentials to access a site. Vulnerabilities include any forms that allow users to POST data. In order to protect from these attacks, we used CSRFProtect from a python module. This creates a token when rendering forms. These tokens need to be included in requests otherwise the request is rejected.

XSS can occur because we are allowing the user to input various data. Jinja does an ok job at mitigating this attack because they escape all values. However, jinja does not protect against attribute insertion which was avoided in this scenario by completely avoiding placing jinja in attributes. We also used wtforms to handle all the form data.

In order to protect to create a more general protection against more XSS or other attacks that may come from different addresses, we implemented a content security policy where the website will only accept scripts from valid sources. So in this case, we are only accepting scripts from our own website. This was done by adding a content security policy http header that defines where to accept scripts from.