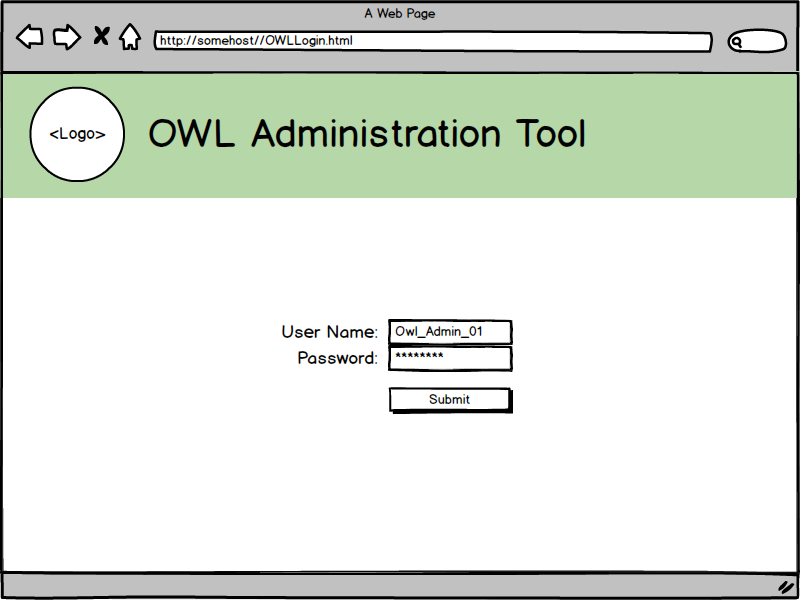
**OWL Login Architecture**

When a user first opens the OWL administration page, they should see a login screen similar to Figure 1:



**Figure 1: OWL administration login page**

When the user presses the “Submit” button, username and password information entered by the user will be passed to the server via a JSON object. This object will have the same structure as a typical query; however, it will be empty except for the username and password information.

When the server receives an “empty” query, it will parse the username and password fields out of the JSON object and compare them against a small set of predefined usernames and passwords stored on a server-side database.

If the username and/or password do not exist in the database, the server will send an error message to the front end. The front end will indicate to the user via an alert that their credentials are invalid, then return to the login page.

If the username and password exist in the database, the server will send a command to the front end redirecting it to the main administration page. As it redirects, the login page will also create a cookie storing the entered username and password information. This data will be used to verify subsequent insertion/deletion calls sent by the administration page.

**Potential Issues**

* Encrypting username/password combinations in cookies, the back-end
* Need to obfuscate URL for Administration page, so it is not accessible by bypassing the Login page
  + Administration page may check for existence of login cookie when it is accessed
  + Administration page redirects user to the Login page if the cookie either doesn’t exist or fails a timestamp check.
  + Alternatively, use IIS to obfuscate the Administration URL?