- Build a web site using previously developed *user web-services*.

- Use mustache templates.

- Use axios as HTTP client to access web services.

- Site available at *<http://zdu.binghamton.edu:1236/users>* (internal network only).

Home Page Links which allow listing all users, creating a user and searching for users.

Users List Page Shows summary of all current users; fields include id, first-name and last-name. Each user has a details button to display further details for that user.

User Create Page Successful creation results in details page for created user.

User Details Page Shows details for one or more users. Each user has a delete and update button.

User Update Page Allow modification of most details for a particular user (does not allow modification of id fields). Successful update results in details page for updated user.

User Search page Allows searching for users by id, firstName, lastName, and email. Will display details of all matching users.

- Make calls to remote web services using axios.
- Catch and rethrow exceptions after extracting error information returned by remote web services.
- *Wrapper code*

Create and update actions use following Post-Redirect-Get form submission pattern:

1. Form is displayed.
2. User submits form using POST. If submission contains errors, form is redisplayed with errors. Else go to next step.
3. Send a redirect to browser to success URL (can be determined dynamically).
4. Browser issues a GET to success URL.

Disadvantage is that any parameters for success page need to be sent as query parameters in success URL.

- Validate almost all input.
- Usually redisplay form on error retaining widget values.
- Distinguish between errors specific to a particular input widget and general errors not specific to a particular input widget.
- Regard all web service errors as general errors.
- Display general errors at top of form.
- Display widget-specific errors next to widget.
- Driver, *Main code*