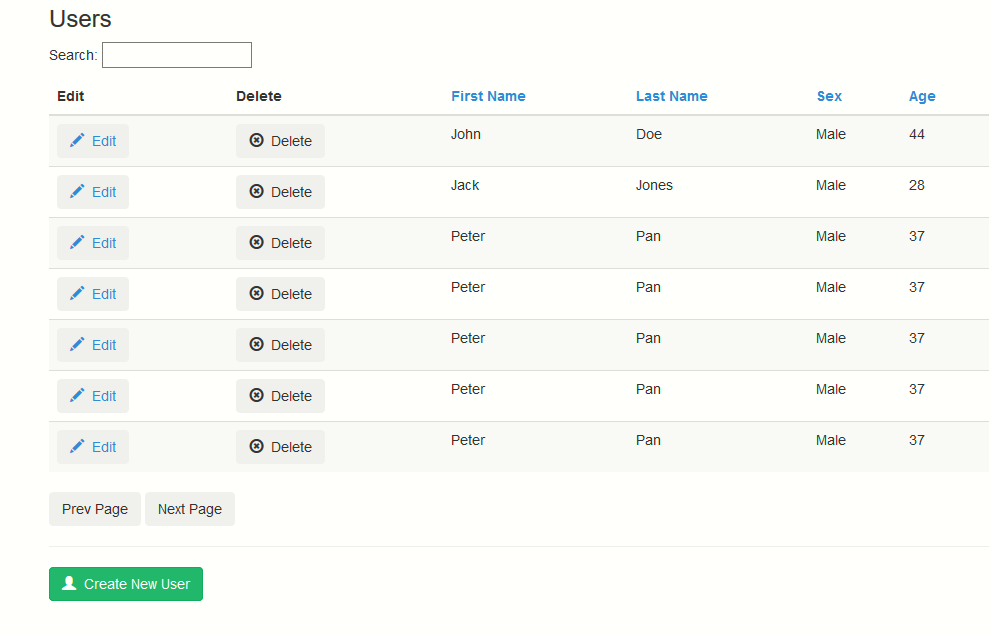
# Project – User List

This project is to build a user list with the ability to add, edit, and delete users with full MERN stack, including a database.

## User Interface

UI should be similar to following:

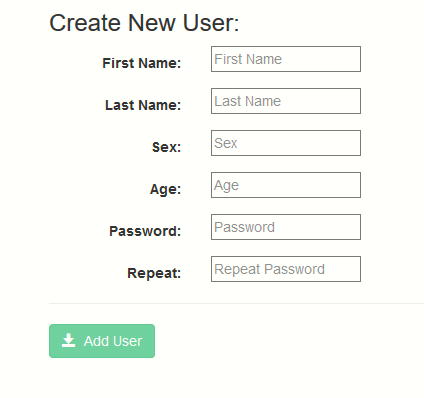
1. User List – list of all active users with Search, Sort (by column with up and down arrows), Create, Edit, and Delete functions



Notes:

* Sort is by column (clicking column header) and has to show either up or down arrow
* Pagination needs to be individual page numbers, not shown in the image, and has to be content sensitive

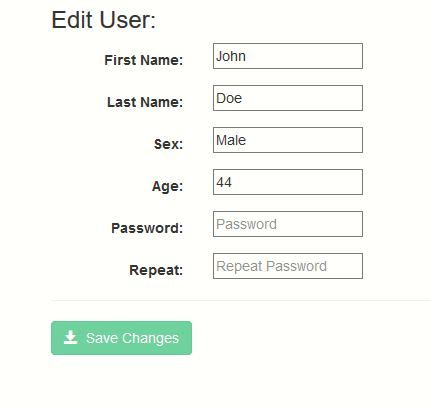
1. Create New User – creates a new user with required information.



Notes:

* Some information will be required, and others are optional. Need to be visible
* Password and Repeat (Password) are for password and have to match
* Use proactive validation to show errors in line
* If the whole form is not valid, the ‘Add User’ button should be disabled

1. Editing an existing user



Notes:

* The user information has to be shown when the page is rendered
* Proactive validation as in New User page

## Functionality

The app should have following functions:

1. When the app starts, it displays user list as shown above.
2. On the list, Search and Sort can be performed.
3. When ‘Create New User’ is clicked, another page is opened to replace the user list, which is a form to allow a new user to be created. After ‘Add User’ is clicked, the new user will be added, this component is closed, and the new user appears in the user list.
4. When ‘Edit’ on a user is clicked, another page is opened to replace the user list, which is a form to allow an existing user to be edited (with current information populated). After ‘Save Changes’ is clicked, the existing users is updated, this component is closed, and the updated information appears in the user list (above).
5. When 'Delete User' is clicked, the selected user will be deleted from the list (and the JS array).
6. Implement paging / pagination by having 'Prev Page' and 'Next Page'. (the size of each page is pre-defined, such as 20).
7. Proactive validation should be performed on both forms. For example, for every action to be performed, values of two password fields have to match.

## Technology Requirements

* Front End
  + Possibly a UI library such as Bootstrap or Material UI
  + HTML, CSS, and JavaScript
  + React as the UI layer
  + Redux as the single state in front end
  + Thunk as the server access
  + Please take care of async data loading!
* Back End
  + REST API
  + ExpressJS and NodeJS server
  + MongoDB with possibly following schema:
* User object should at least include these properties:
  + \_id - default
  + First Name
  + Last Name
  + Sex
  + Age
  + Password
* User authentication is optional.