Architectural Patterns

**DEADLINE:** 04/11/2018

## FOLDER STRUCTURE

FL\_9\_12\_homework\_architectural-patterns/\*

   homework/\*

       src/\*

data.js\*

index.html\*

index.js\*

style.scss\*

.eslintrc.json\*

package.json\*

package-lock.json\*

webpack.config.js\*

## PREREQUISITES

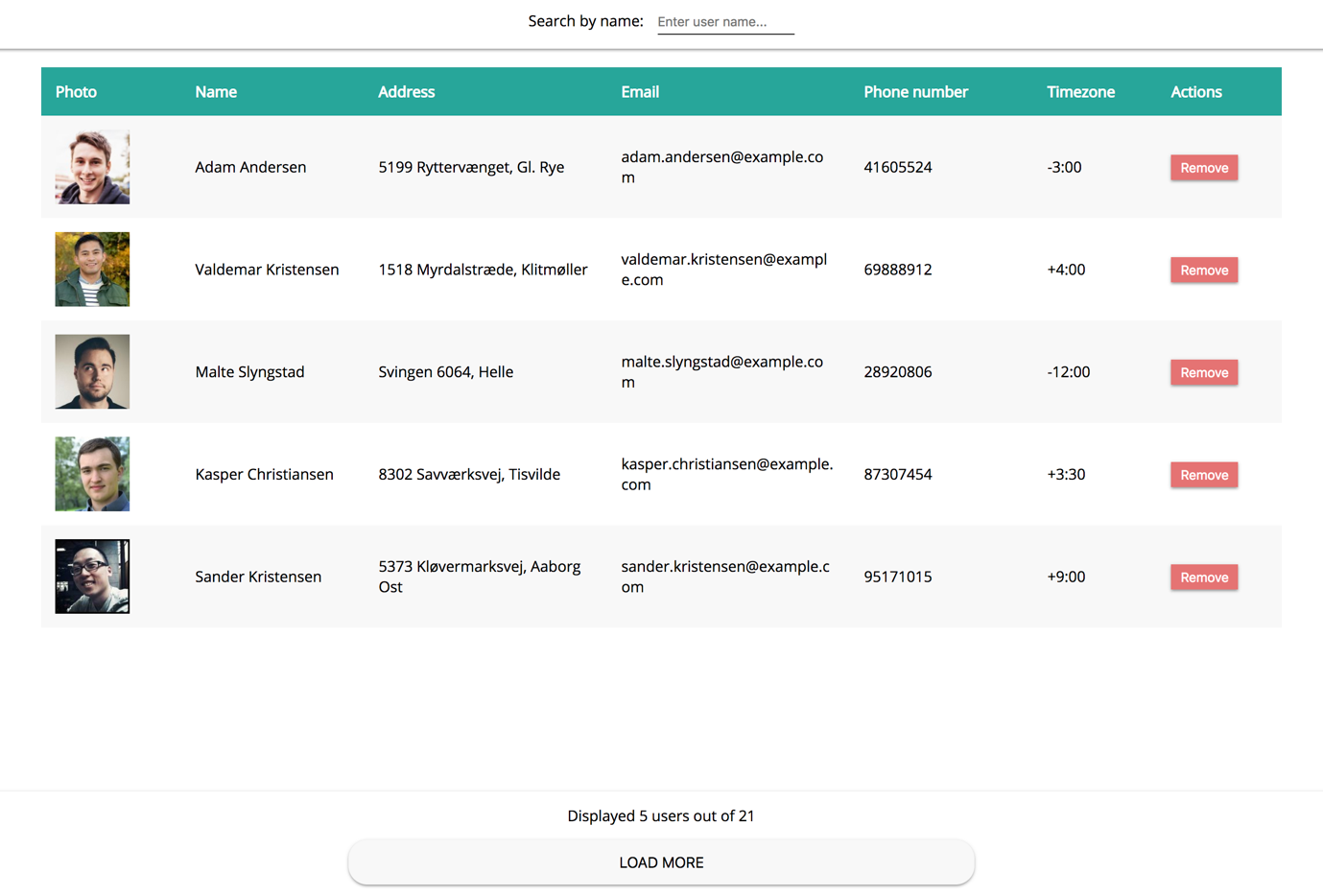
Requires `node.js` and `npm` to be installed.

## HOW TO RUN THE PROJECT

1. Download homework
2. Extract files to your folder
3. Open cmd (terminal, bash)
4. Navigate to your folder which contains package.json file
5. Run `npm install` command
6. After all modules have been installed, use `npm start` command to run local server with project.

## TASK

Your task is to create a simple redux application. It should consist of one page, which contains information about some non-existing users (see Figure 1). There should be an ability to search users by name, remove them and load more users on button click.



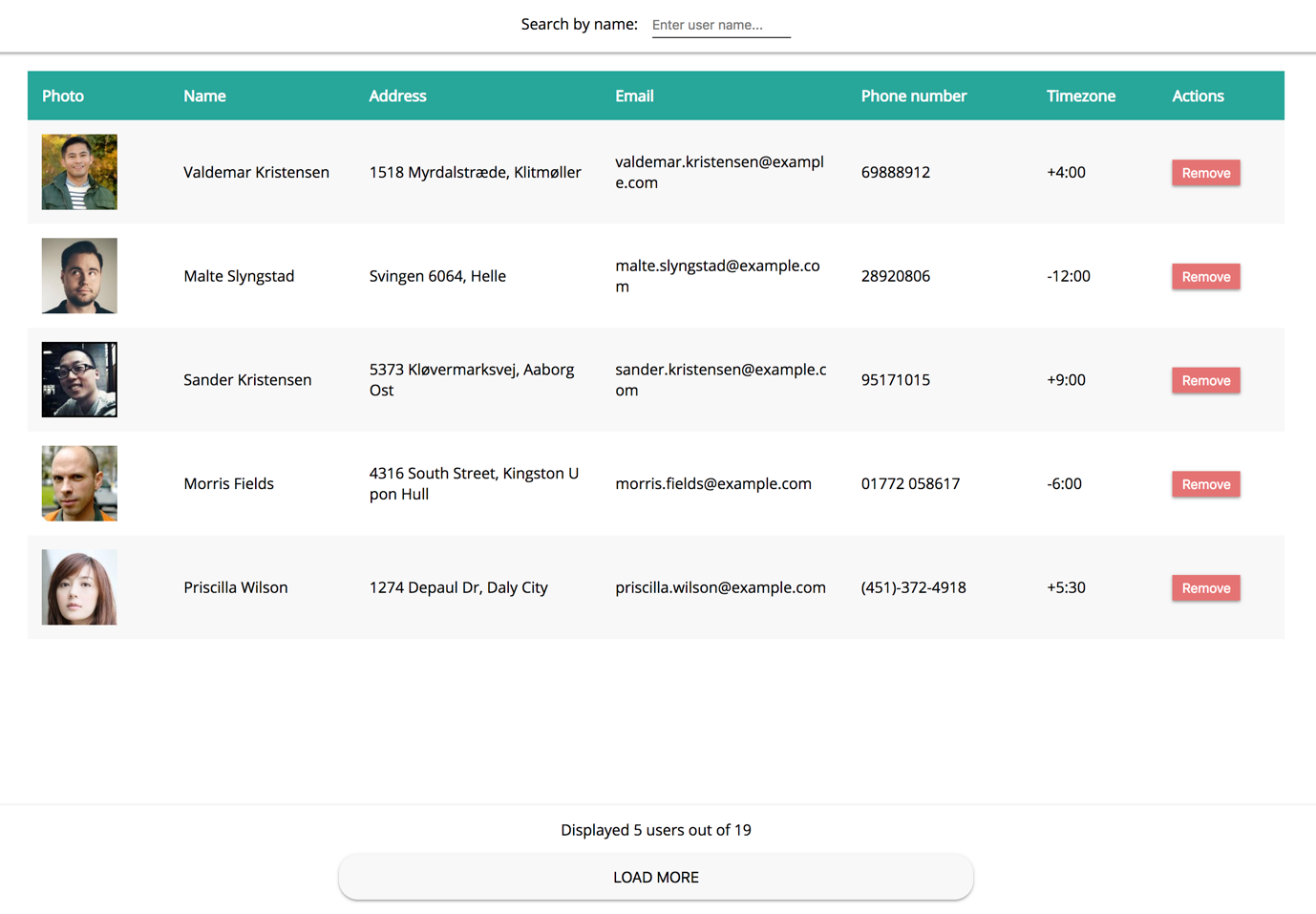
##### Figure 1 — Initial users page view

Page should contain three main components:

* Users table
* Users count info & load more button
* Search input

1. Users table

Initial table size should be 5 rows. When user click on remove button, amount of users should be reduced, and users table should be updated (see Figure 2. In that case users Adam Andersen and Kasper Christiansen were removed). If all users will be removed, appropriate message should be displayed (e.g.` No users are found `). You can import user data from */src/data.js* file.

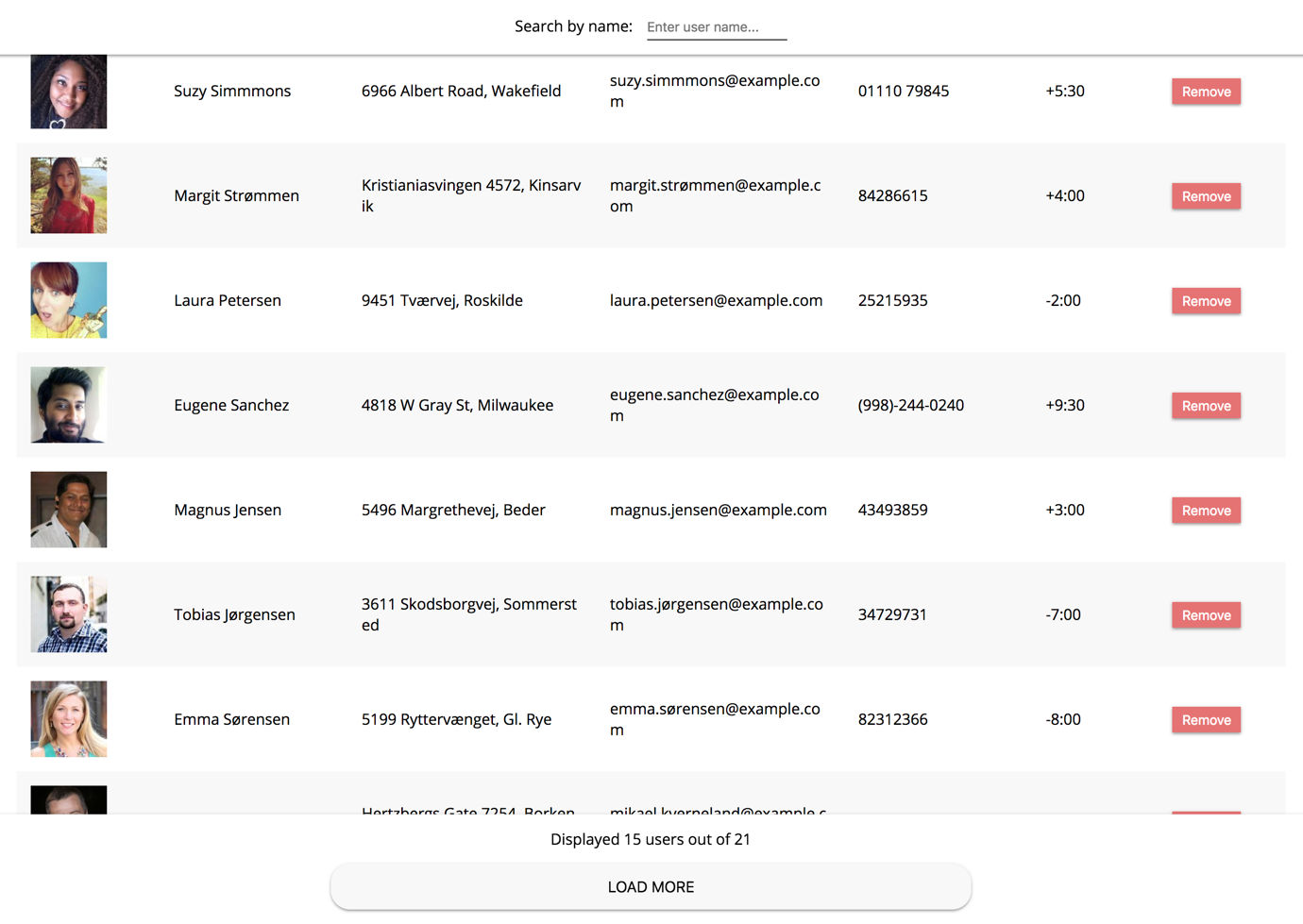


##### Figure 2 — View after users removal

1. Users count info & load more button

When user click load more button (see Figure 3) users table size should be increased by 5 rows (or less, depends on how much users are left). Note that users count info & load more button and search should always have fixed positioning. If there are no users to load, load more button should be hidden.

User count info should contain number of displayed users and total amount of users (total amount of users depends on number of removed users or search query).

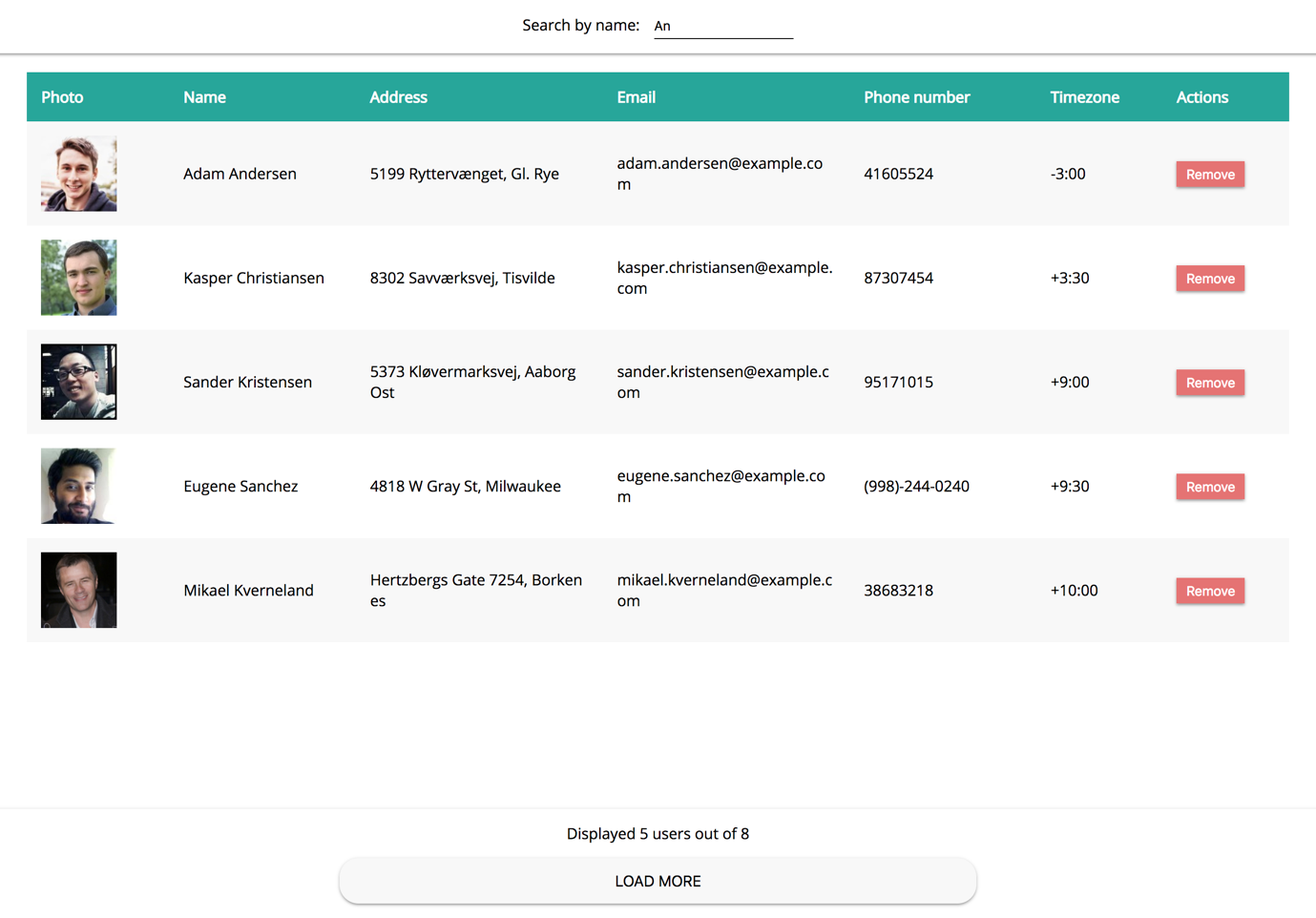


##### Figure 3 — View after load more actions

1. Search input

User should have an ability to type search query in text input and get following results:

* Users table and count info should be updated on each search query change
* Search query should be used to find users by name property
* After every query change the total amount should be equal to number of found users by this query
* If total amount of users (users found by search query) will be more than 5 users, table should contain 5 user rows (same as on initial view) and load more button should be available. If <= 5 uses, load more button should be hidden. And if total amount of users will be 0, there should appear an appropriate message.



##### Figure 4 — Search results

## PLEASE NOTE

* Pixel perfect is not required
* Check video and screenshots in examples folder for better task understanding
* Read this [article](https://www.sitepoint.com/redux-without-react-state-management-vanilla-javascript/) before you start doing task

## RESTRICTIONS

* Do not use any additional libraries/frameworks
* Predefined files (webpack.config.js, package.json, data.js, index.html, eslintrc should not be touched at all)
* Do not commit node\_modules folder

## HOW TO

* To run linter for the project use `npm run lint`

## BEFORE SUBMIT

* Remove all unnecessary files that you might have included by mistake
* Verify that all functionality is implemented according to requirements
* Add comments if the code is difficult to understand
* Fix warnings/errors in the browser console
* Verify that the name of the folders and files meet the requirements
* Make sure there are no errors/warnings in the browser console
* Run the linter and fix all warnings and errors

## SUBMIT

* The folder should be uploaded to GitHub repository 'fl-9' into master branch

## USEFUL LINKS

* [https://www.sitepoint.com/redux-without-react-state-management-vanilla-javascript/](-%09https:/www.sitepoint.com/redux-without-react-state-management-vanilla-javascript)
* <https://redux.js.org/>