# [WebTraining3: Javascript part 1](https://confluence.app.iaf/display/HAD/WebTraining3%3A+Javascript+part+1)

# Exercise:

Now we will add some functionality to our app.

The following exercises should be done with [javascript DOM manipulation functions](http://vmd-jtlsd1.srv.base0324.iaf/content/books/Web/Javascript/Eloquent%20Javascript/Eloquent%20JavaScript/chapter12.html).

# Tweeting

After clicking the tweet publish button, the text box should be erased of it’s content, and a new DOM element with the text should be appended to the form a list of your tweets.

Until we’ll have a server side, we’ll load some data from a local JSON array for our tweets list, instead of the static HTML code, for example:

|  |  |
| --- | --- |
| 1  2  3  4  5 | [   {username: 'Bobo', text: 'hello followers!'},   {username: 'Elvis', text: 'this exercise is really easy!'},   {username: 'Mimi', text: 'I want to go to sleep'}  ] |

Use this data for the list of tweets of the people you follow on the main page. Make sure that your new tweet is pushed to the JSON array. You may hardcode your own made-up username.

# Users page

Implement the functionality of the ‘Follow’ button:  
The button has 2 states: ‘Follow’ and ‘Unfollow’. Clicking the button toggles between these 2 states, and changes the color of the button too.

Also, add a list of people you follow (in the users page). So when you toggle between states, their avatar and name is added / removed from the list.

Save the users list and the folowees list in a local JSON array, similar to the tweeting functionality.

The name filter text box should actually filter the users list while typing. That means that if you begin typing in the box, and you wrote ‘El’ – only users with names containing ‘El’ should be visible on the users list (like Elvis or Elai). When you add ‘v’ (resulting in ‘Elv’) only users with names containing ‘Elv’ are visible in the users list (like Elvis. Elai is filtered out).