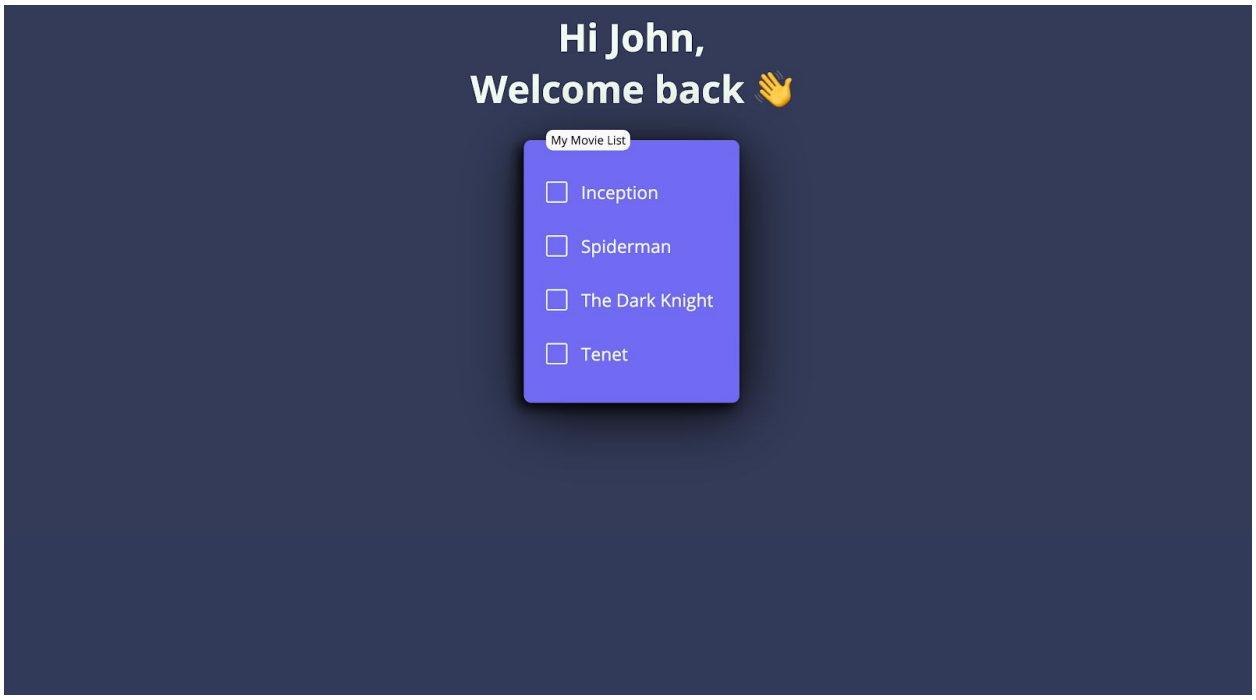
Important:
In app.js within the starter, at the very top in a common, please specify who your partner's names and student numbers are.

Creating a dynamic movie list web app:

EJS Cheat Sheet for your reference:
https://www.notion.so/EJS-Cheat-Sheet-5e1725095ed748e0b9d3e99be7876f7b

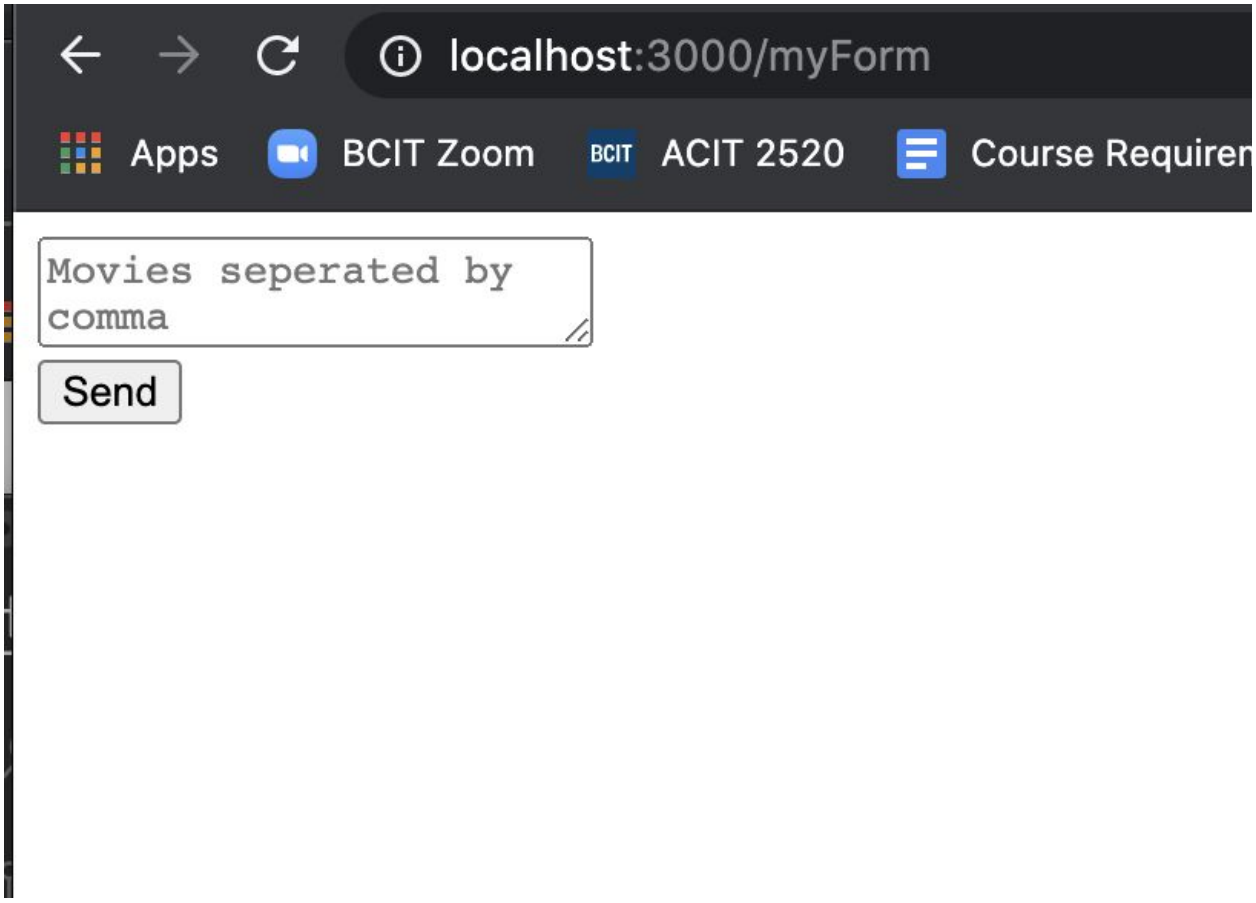
Download the starter.zip file, and extract it to your computer. Run the application and visit localhost:3000. You will see the following:



The data above is completely static data. It will soon be completely replaced with dynamic content. For now, go to index.ejs in the views/pages folder and you'll see this data hard-coded into the HTML.

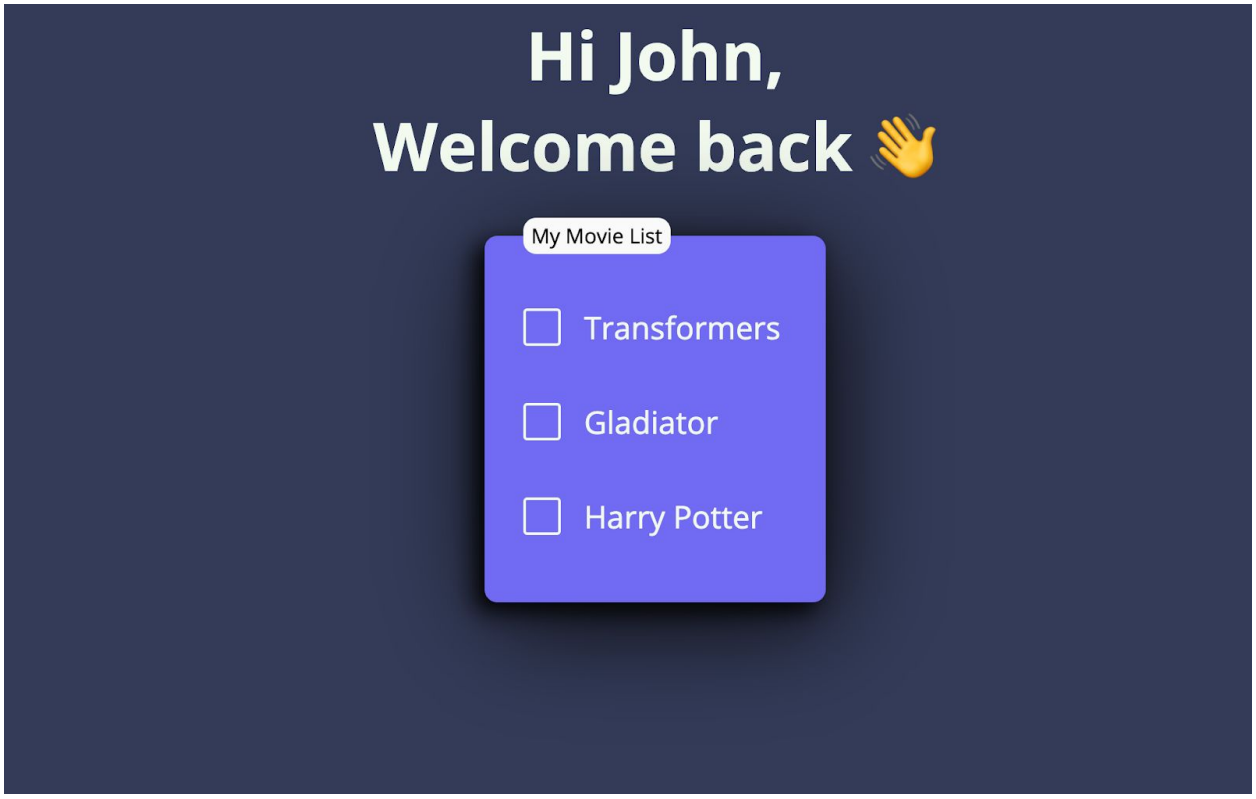
For this Lab, you must do the following:

When I visit (GET) the localhost:3000/myForm URL:



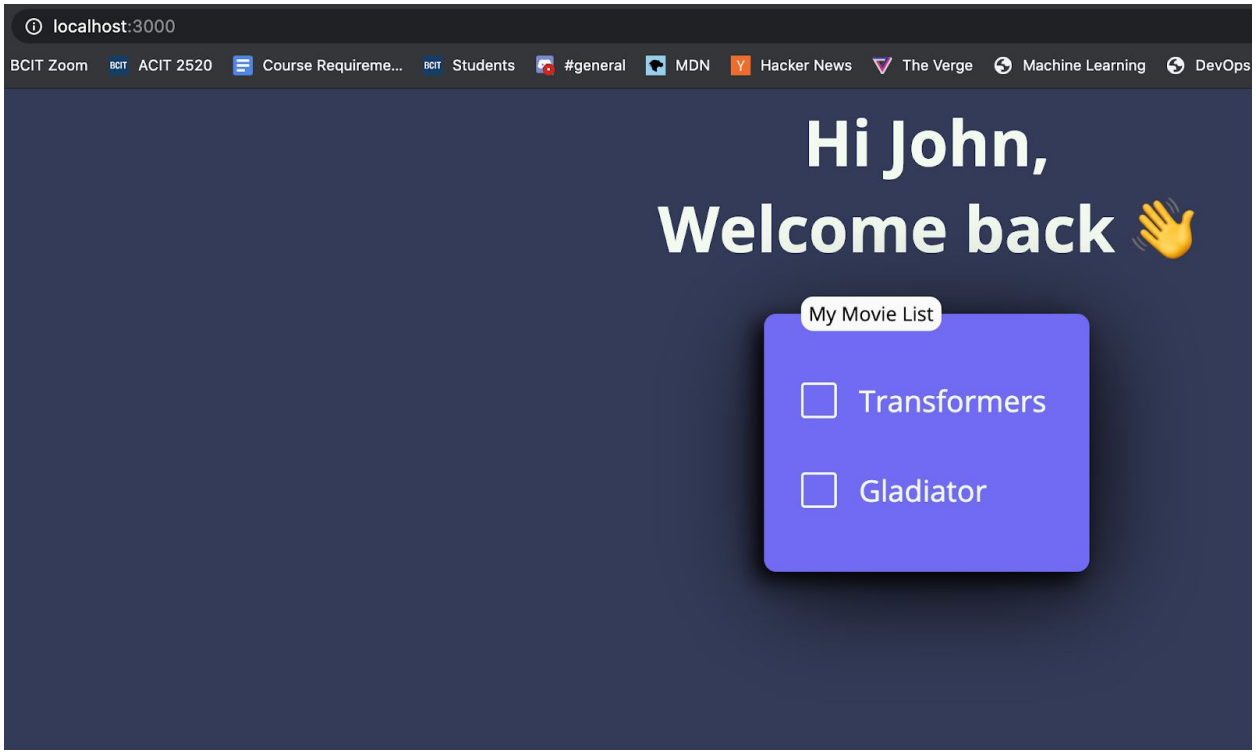
I should be able to see a textarea, which allows me as a user to insert comma-seperated movie name items.

When the user clicks send, a POST request to /myForm should occur. Inside the express server for this, you should handle the response by taking those todo items obtained from the text area and render them (display them) in the views/pages/index.ejs page. This means that instead of seeing the hard coded movies from before (Inception, Spiderman, The Dark Knight, Tenet), we should see (If I typed into the textarea : Transformers, Gladiator, Harry Potter):



Next, if I visit: http://localhost:3000/myListQueryString?movie1=transformers&movie2=Gladiator

The browser should make a GET request to your server's /myListQueryString function, and it should handle the response by taking the query string parameters and sending them, again, to views/pages/index.ejs. You may assume there will only be 2 query string parameters, with the names movie1 and movie2. You may assume they won't be left blank.
Expected Result:



Lastly, if I visit: http://localhost:3000/search/transformers

The browser should make a GET request to your server's /search function, and this function should read through the text file: movieDescriptions.txt, checking to see if there is a movie description for the movie passed after /search/ in the :movieName query parameter. If the movie does exist in the text file, pull out it's description and put it inside the views/pages/searchResult.ejs page. The page should contain 3 things:

1) An H1 tag containing the name of the Movie
2) A p tag containing the description
3) An <a> tag link that sends us back to the home page.

If the movie does NOT exist, simply show a message on searchResult.ejs saying: "Movie could not be found".

Extra practice:
If you finish all of the above, try making the name "John" dynamic. In other words, create a variable that contains your name, and then replace John with your name.