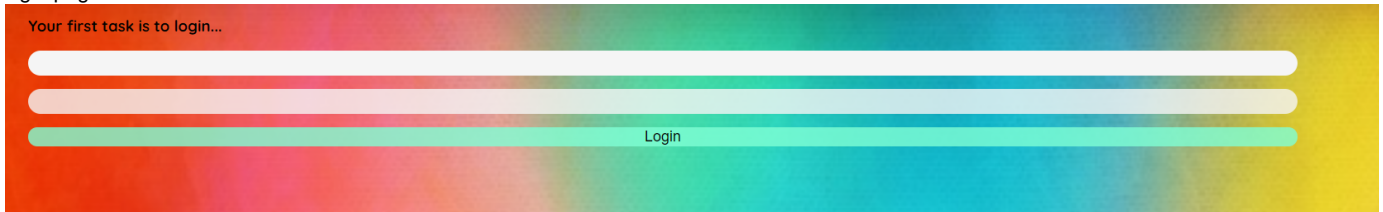


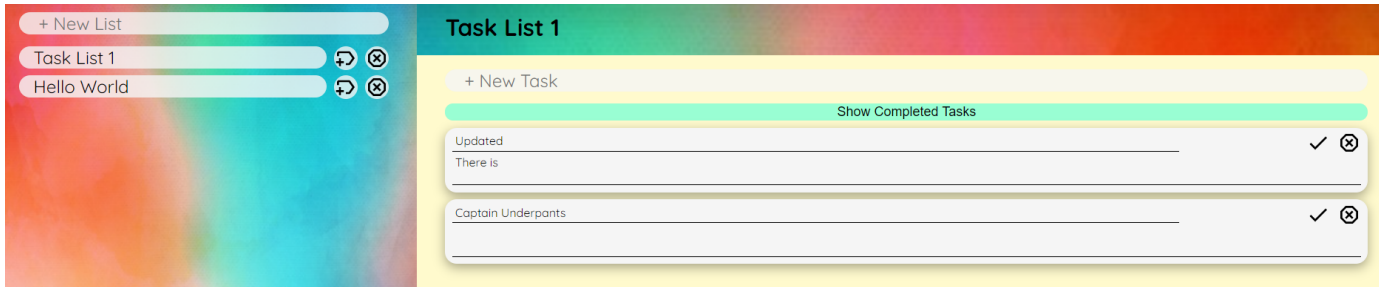
## Day 4

Today I have began to implement the front end for the project...

login page:

The login page features a vibrant, multi-colored background with a gradient of red, orange, yellow, green, and blue. At the top, the text "Your first task is to login..." is displayed. Below this, there are three horizontal input fields. The bottom-most field is a green button labeled "Login".

tasks view:

The tasks view is divided into two main sections. On the left, there is a sidebar with a list of task lists: "+ New List", "Task List 1", and "Hello World". Each list item has a circular icon with a plus sign and a circular icon with a minus sign. The main section on the right is titled "Task List 1" and contains a "+ New Task" button. Below this, there is a "Show Completed Tasks" button. The task list itself contains two items: "Updated There is" and "Captain Underpants". Each item has a checkmark icon and a circular icon with a minus sign.

Now you can see why it is called tie-dye!

Tasks can be grouped into different "Task Lists" (similar to Google Tasks or Microsoft ToDo)

I faced several hours of difficulty trying to authenticate to Keycloak from angular. Keycloak was stating that CORS was not supported, however this was not actually the issue.

I had to refactor my Angular code to use x-www-form-urlencoded content-type in order for it to work correctly.

All the code is in the git repos which are linked from the main page

```
login(username: string, password: string): Promise<boolean> {
  return new Promise<boolean>((resolve, reject) => {
    const options = {
      headers: new HttpHeaders().set('Content-Type', 'application/x-www-form-urlencoded')
    } as HeaderOptions;
    let body = new URLSearchParams();
    body.set('client_id', environment.clientId);
    body.set('grant_type', AuthService.GRANT_TYPE_PASSWORD);
    body.set('username', username);
    body.set('password', password);

    this.http.post<AuthResponse>(environment.KeycloakAdminBaseUrl + this.authEndpoint, body.toString(), options).subscribe(
      response => {
```

Still To Do...

I still need to implement the complete functionality and the show/hide complete functionality. this is already completed in the back end but not implemented from angular.

Also I need to correct some styling problems

If I get time, I would also like to implement:

- Task ordering: by priority
- Parent and child tasks