1.) **Team number:** 015-04

2.) **Team members:**

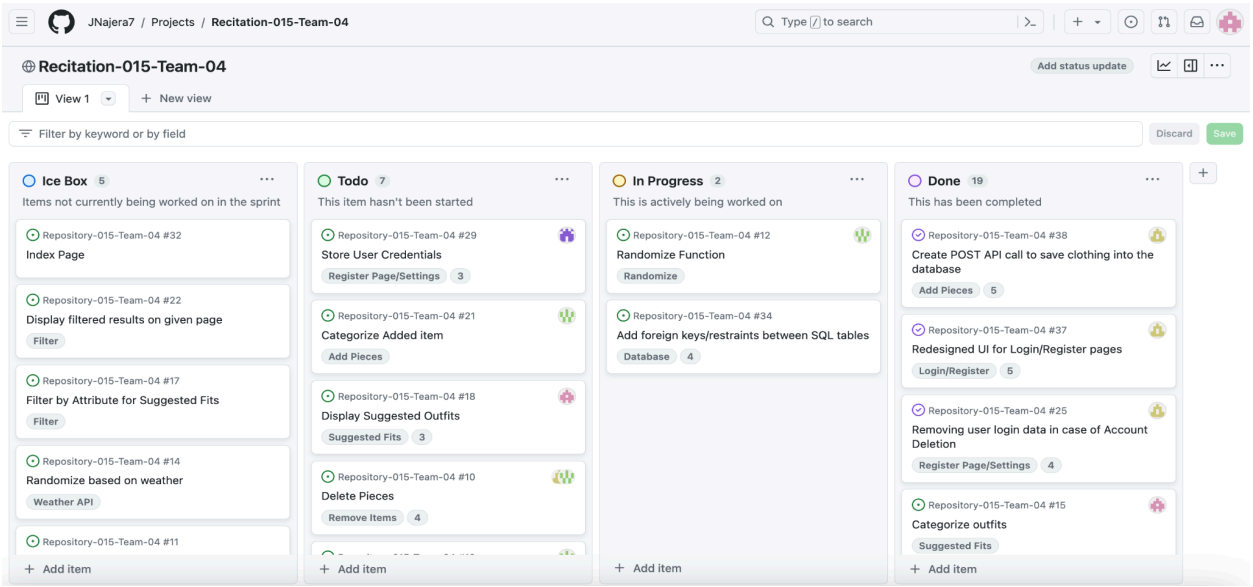
- Jesus Najera
- Parker Taranto
- Darwin Ortiz Garcia
- Elsa Haynes
- Kevin Shu

3.) **Project Description:**

Wardrobe Wizard: Style made easy.
Spend less time deciding what to wear and let the Wardrobe Wizard decide for you! Once registered for an account using our login system, each user can upload articles of clothing (“pieces”) into their digital closet. These pieces are then organized by category and displayed to the user in the Saved Pieces page, where the user can manage and delete them. From there, the user can use our outfit designer to add their favorite combinations of clothing into the Saved Fits page. As a bonus, the home page displays the current local forecast, which gives context to any outfits made based on the weather.

4.) **Project Tracker:**

https://github.com/users/JNajera7/projects/1



5.) **Video:** https://youtu.be/XLKZHhRAkCE

6.) **VCS:** https://github.com/JNajera7/Repository-015-Team-04

7.) **Contributions:**

Jesus: Designed and created the login/register pages, created various API calls in the index,js file, and set up the portion of the database that stores the users wardrobe. Some of the API calls that I made include the deletion of user data in case of account deletion and some of the addnewpieces call.

Parker: Worked on the saved outfits page, designing the layout and functionality. Contributed to the general homepage layout.

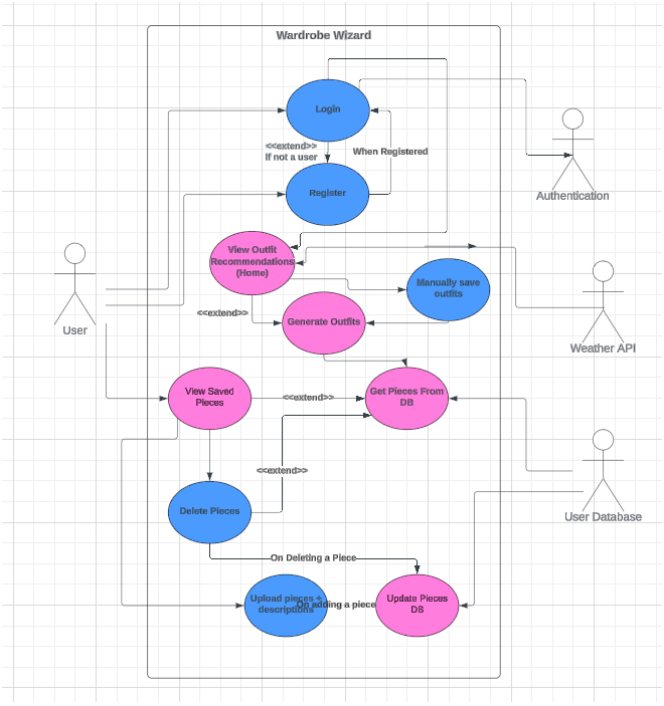
Darwin: Worked on the delete pieces pages, added the password requirement feature, and contributed to the navbar.

Kevin: Worked on the adding, saving, and displaying of pieces/fits, as well as the weather API and homepage. Made the /savedpieces, /savedfits, /addpiece, and /addfit API calls using Node.js. Also set up docker and deployment steps, such as database initialization.

Elsa: Worked on general template for the savedpieces page, directed the database layout, and contributed to adding pieces (form + backend).

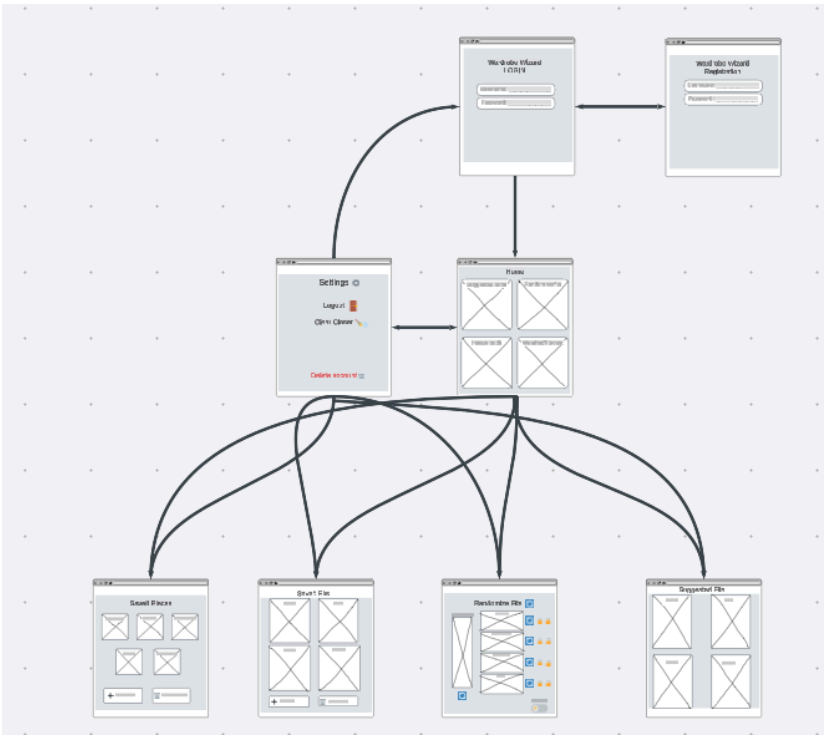
8.) **Use Case Diagram:**

https://lucid.app/lucidchart/936e30a8-5589-425a-8e02-d8347129fd11/edit?viewport_loc=-1452%2C-1281%2C3399%2C1585%2C0_0&invitationId=inv_cb7b7bf0-5c7f-4580-8788-0317fc349d69



9.) Wireframe:

https://lucid.app/lucidspark/e083f29c-6f8b-423e-beac-c422b9c04753/edit?invitationId=inv_267aa81a-9530-4fcc-bfdd-e2407305dca4



10.) Test Results:

- **Testcase 1: Testing registration**
  i.) Input: {username: 'JohnnyDoe', password: '197254Th_'}
  ii.) Expect: redirects to '/login' and res.body.message == 'Success'
  iii.) Result: This test case passes and redirects the user to the login page.
- **Testcase 2: Testing duplicate usernames**
  i.) Input: {username: 'JohnnyDoe', password: 'password'}
  ii.) Expect: res.status == 400
  iii.) Result: The test case attempted to register with a username that is already in the users database and is unable to.
- **Testcase 3: Testing login page rendering**
  i.) Expect: redirects to '/login' and status == 200
  ii.) Result: This test case passes and redirects the user to the login page
- **Testcase 4: Rendering nonexistent 'log' page**
  i.) Expect: attempts to redirect to non-existent '/log' endpoint and status == 400. Then redirects to the login page by default.
  ii.) Result: This test case passed and redirects the user to the login page.

11.) Deployment:

Use command `docker compose up -d` in the ProjectSourceCode directory after cloning.
http://recitation-15-team-04.eastus.cloudapp.azure.com:3000/login