1. **Project Title**: ***CampusCache***

2. Project Summary:

- **Project Description:**
  We want to create a portal for students to use in order to reserve various resources from various locations. This would solve the problem of not knowing if a resource is in use and finding where they are available. With a good scheduling policy this could even solve the issue of distributing resources fairly to give students an equal chance to use them.

- **Creative Components:**
  - One creative component would be the web application deployment on a Kubernetes cluster. Kubernetes would be a technically challenging component of the project that would merit important features required for a robust application including fault-tolerance, cost-effectiveness, deployability across multi cloud environments, and scalability. One way we are planning on achieving this is with SkyPIlot. SkyPilot is a framework that will help automate the heavy-lifting of deployment, including the provisioning of a cluster across multi cloud nodes.
  - Another creative component that can potentially improve the functionality of our application is to display available desktop computers at a certain region of a facility by hovering over the region of interest. For instance, if a user is interested in looking for available double monitor desktops on the 4th floor of Grainger Library, we can display a floor plan like such:

  Then, we can have our app display the number of available desktops when the user hovers over the EWS Computer lab with his/her cursor like the image above.

- **Usefulness:**
  When borrowing things such as chargers or other materials from Grainger or any other building, to our knowledge, the current system is outdated and inconvenient. Through our website, one can immediately check what items are available to be borrowed, rather

than going in-person and asking. Currently, after we scan our netID, the borrowed item is tracked on a piece of pen and paper. Due to human error, this can lead to many items not being tracked properly. Our app will streamline this process, automating this interaction, enhancing convenience and ensuring that the items are returned. Also, if a student requires an item to be borrowed for a course or another reason, a professor will be able to reserve the item for the student.

- **Realness:**

  For our data, we would require a dataset for users, course registration, and for the resources. Our users dataset would consist of netIDs and major(s). The users dataset would need to have over 55,000 rows to accommodate the number of students. The course registration dataset would have netIDs and course CRN. The course registration dataset would probably have over 200,000 rows assuming each student is taking 4 courses. And lastly, the resources dataset would have the resource name (i.e. computer, soldering station, etc), it's status (reserved, free, in use, out of service, etc.), building, and identifier. This dataset would have 1,500 rows.

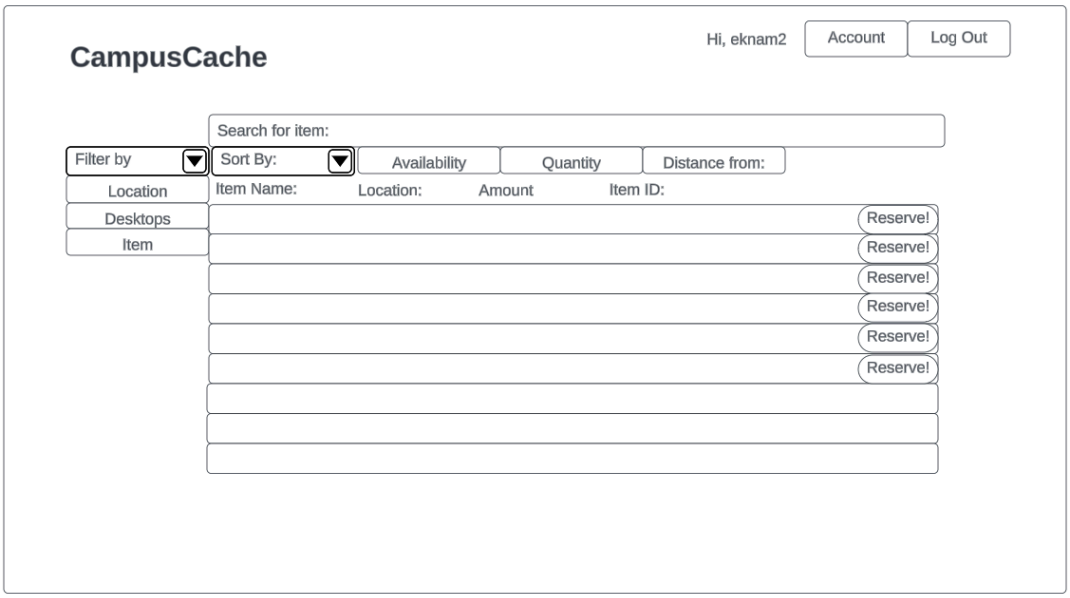
- **Functionalities**
  a. Create
     i. Add new items
        1. Create unique id for each item
     ii. Add new facility
     iii. Reservation
        1. Adding a new transaction
  b. Read
     i. Check real time availability of item
     ii. Display Reservation Queue / Calendar
     iii. Display average time item is used for
     iv. Finding nearest facility a certain item can be borrowed from based on location / distance
     v. Display floor plan of certain areas of facility, and show availability of desktops of corresponding region
  c. Update
     i. Update inventory with CSV files that fit our schema
     ii. Update Item condition
        1. Working, needs to repaired etc.
        2. Available, or being used
     iii. Archive past reservations
        1. Timestamps of each reservation → calculate average time each item is borrowed for
  d. Delete
     i. Reservation
        1. Removing a transaction that's over from the current transaction table

ii. Deleting items

1. Delete items that can no longer be used or provided

Low Fidelity UI Mockup:

- Student UI



- Desktop Availability:

Project Work distribution:

1. UI/UX (design, color theme, library): Elliot

2. Frontend (state management, component logic, requests): Leo
3. Backend (routes, data pipeline, connecting to database): Cristian
    - Data Pipeline (transform, cleaning, streaming)
4. Database: Leo, Luke, Elliot, Cristian
5. CI/CD (containerization, image scanning, deployments, provisioning): Luke
6. Code Reviewers: Leo, Luke, Elliot, Cristian
7. Task Creation: Elliot Nam