**Requirement & Prototype Document**

**Introduction**

Introduction

1.1 Overview

The course resource library is the 1.0 is the first release of the system. The web based library will allow instructors to organize resources, tools, and assignments for students.The goal of this project is to create a user friendly web based course resource library that will enable learners to easily access a curated list of resources, tools, and activities to further their learning journey. In order to use this application users will be asked to sign in or create an account. User accounts will be divided into two groups, learners and educators. Learners will be able to enter the course id of the course they are interested in and educators will be allowed to accept students into the course. Students will then be able to access the curated library to aid their studies. Educators will be able to organize information into categories and subcategories.

1.2 Goals

The goal of the 1.0 system system release is the implementation of the following key features:

1. Login and registration
2. Database implementation
3. Course page creation for educators
4. Ability to create categories and subcategories to organize information
5. Learner access to add a course
6. Educator access to accept learners
7. File management
8. Educators will be able to upload and download files
9. Students will allowed to download files from components and submit into files into assignment components

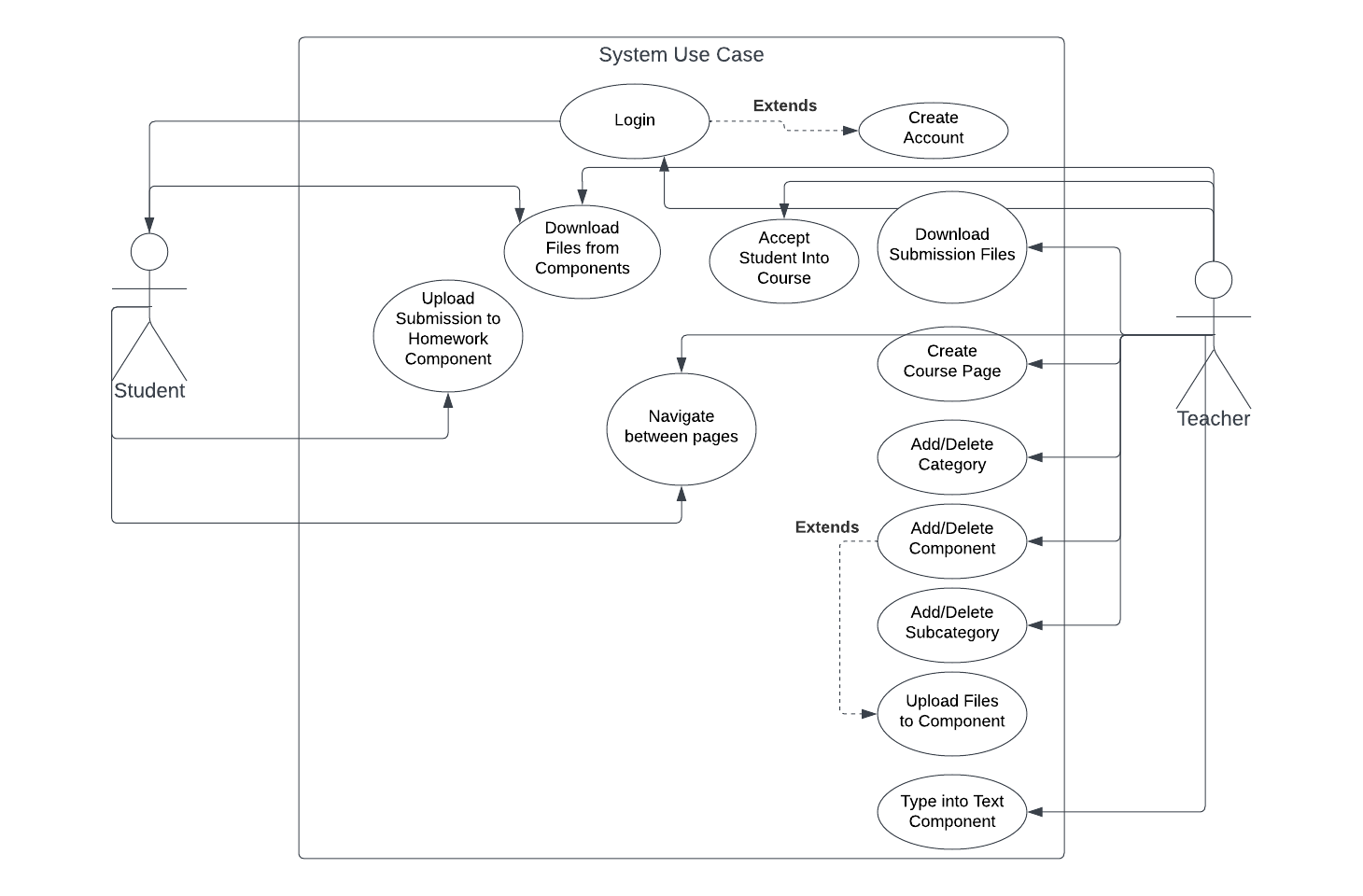
1.3 Constraints

The team has one semester to build a cross browser course resources library and implement key features. Due to the needs of different browsers the team will need to use an array of both browser specific and cross compatible libraries and frameworks. Additionally, the website capacity may be limited as the server will run from a local host.

**Glossary**

1. Database: An organized collection of related data
2. Text Field: An area where people can enter or edit pieces of text
3. File Type: A name given to a specific kind of file

**User Requirements**

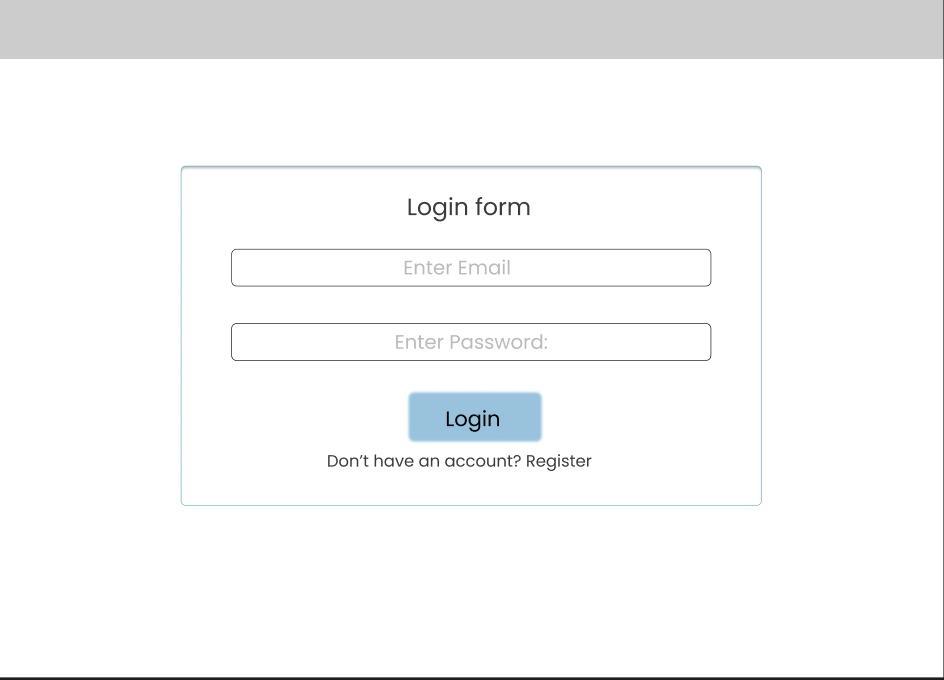
****

**List of Functional Requirements**

1. User can login to a user created account with a username and password or create an account if they don’t have one.
2. Teachers can add or delete a new category and add or delete students to a category as well.
3. Teachers have the ability to add or delete subcategories as well as add or delete components to any category or subcategory.
4. Users are able to upload and download files from components.
5. Teachers can type into text components.
6. User is able to navigate through pages.

**System Requirements**

*Sign in Screen*



The user will log in through this page.

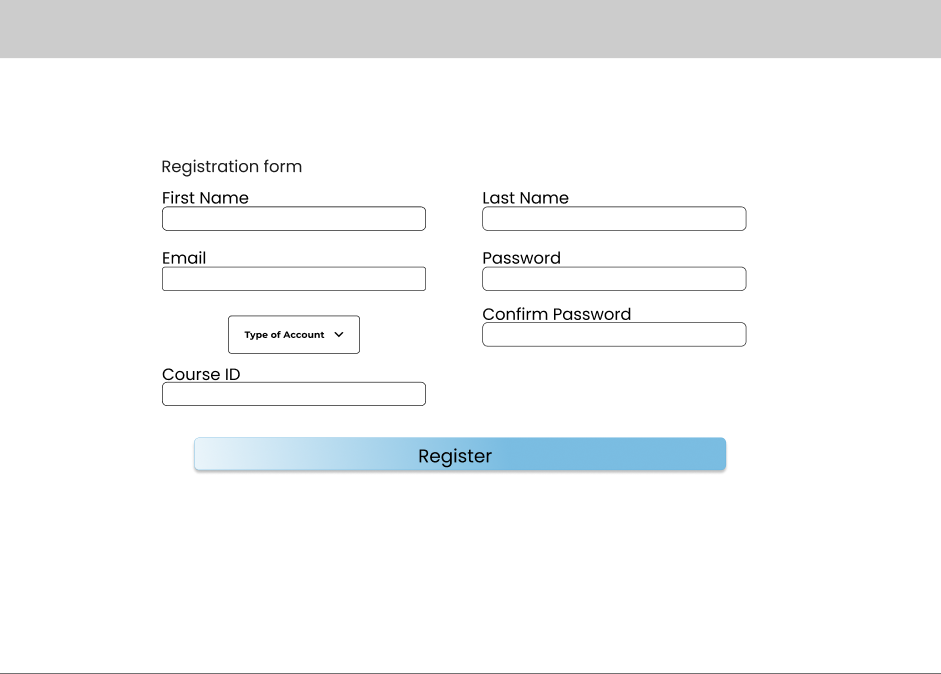
The user can change to the registration page by clicking on the registration button.

Logging in requires the user’s email and password. Clicking the login button will verify if the given email and password has a matching user in the database.

If the user is a student, the database checks to see if they have access to the course they defined when registering.

If these checks are passed, the user will be brought to the course homepage they are a part of.

*Registration Page*



The user can register an account through this page. The account created will allow the user to access the course specified.

Teacher and student are the only types of accounts.

If the account is a teacher, the database will verify that there is not a matching Course ID value in the database currently. If there isn’t a new course with the given ID will be made.

If the account is a student, the database will check to see if there is a course with the same Course ID entered. If there is, a request to join the course will be sent to the teacher of that course.

A completed registration will add the information in the fields into the database.

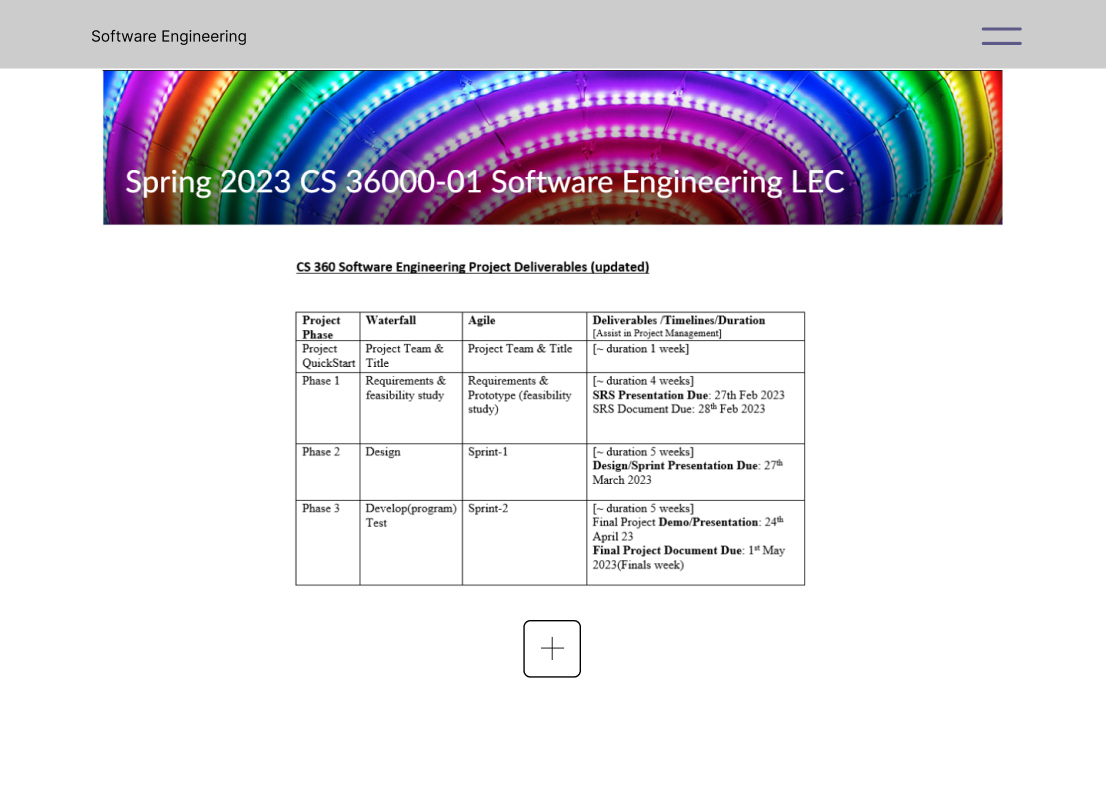
When completed the user will be taken back to the login screen

*Course Home Page*



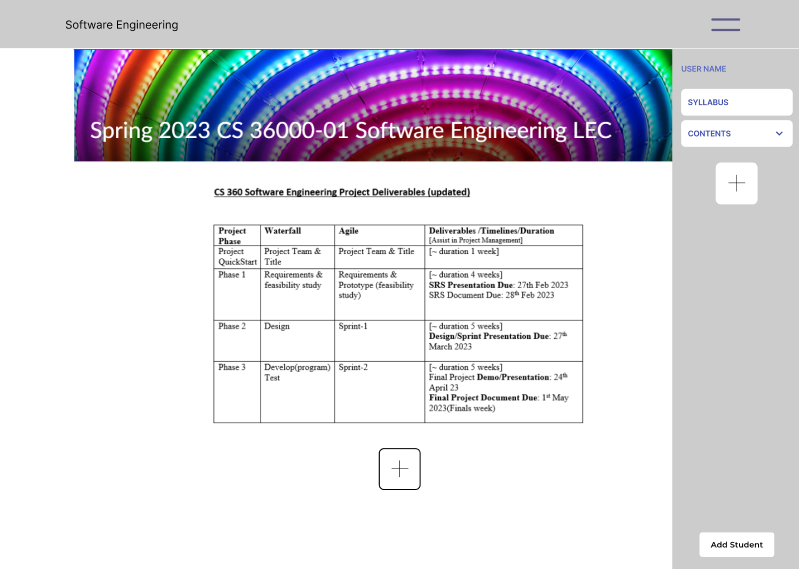
1. Add button 2. Add student button

3. Add subcategory button



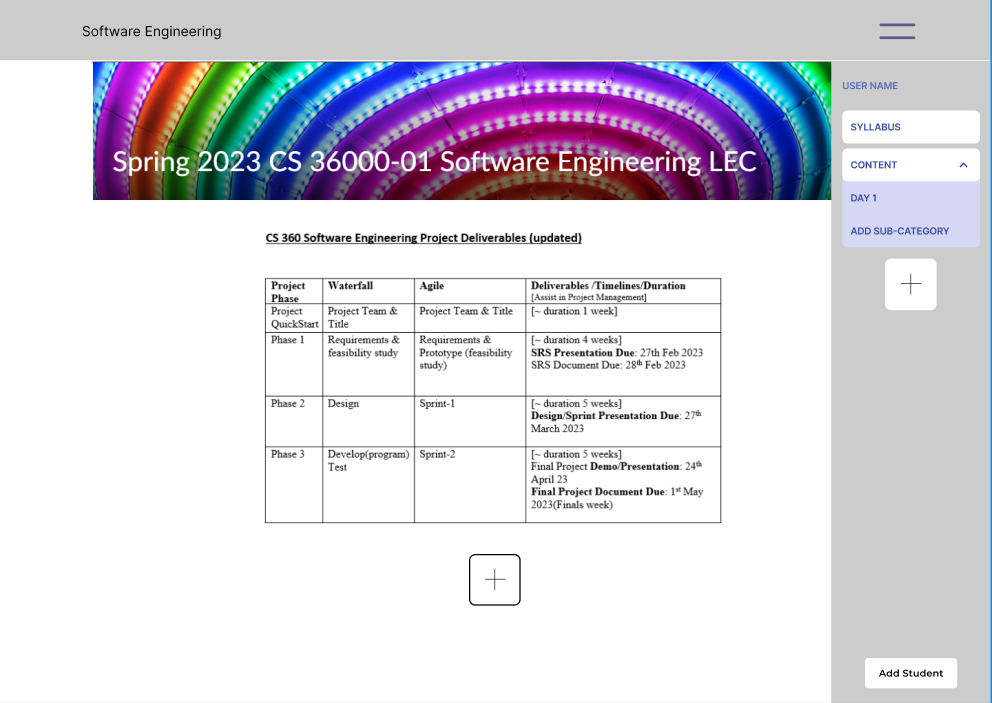
This is the main page of the resource library and is the first page seen after logging in.

Right clicking on a component or a category/subcategory will display a selection allowing the deletion of the component or category/subcategory.



By clicking on the text field in the top left, a teacher can change what the course’s name is.

Clicking on the double-bar at the top right, will open the sidebar containing the user’s first and last name and all the categories/subcategories added by the teacher.



Clicking on a category or a subcategory will bring the user to its page

Clicking on the dropdown arrow of a parent category will display its subcategories.

The add-buttons, add student button, and add subcategory button will only be visible and activatable by a teacher.

Clicking the add button on the page, will bring up the add component popup to the current page. This popup is used to add components to the current page (More detail below).

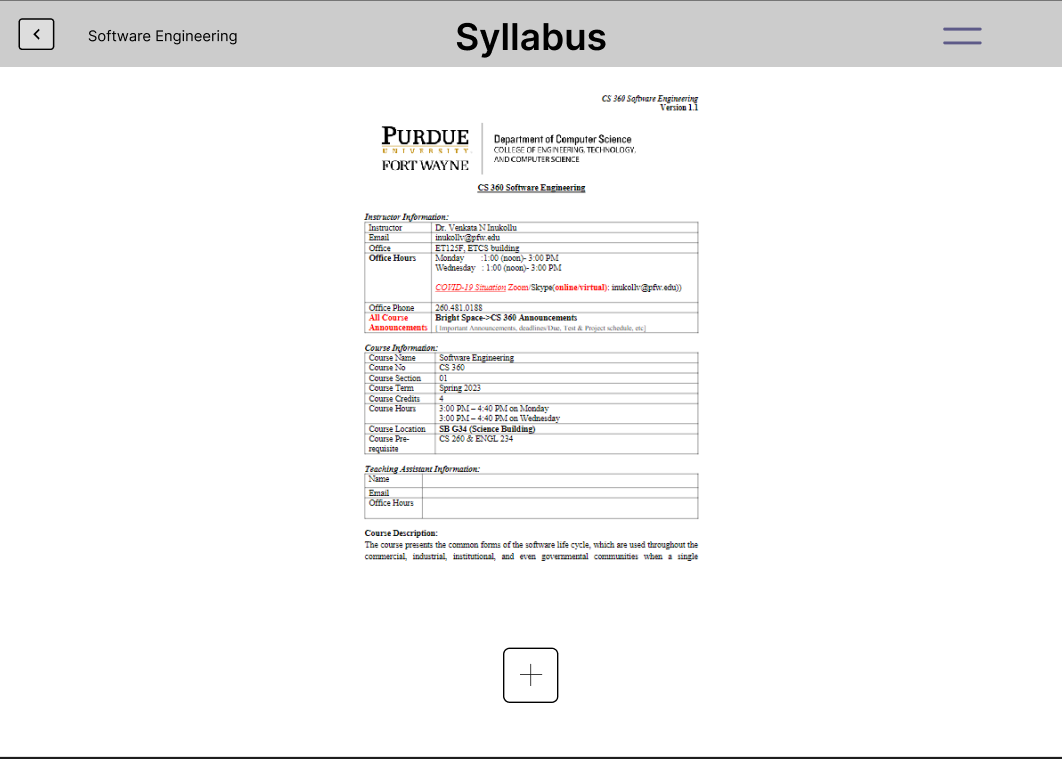
Clicking the add button that is on the sidebar will bring the add category popup to the current page.

Clicking on add subcategory will bring up the add category popup to the current page.

Clicking on the add students button will bring the user to the add students page.

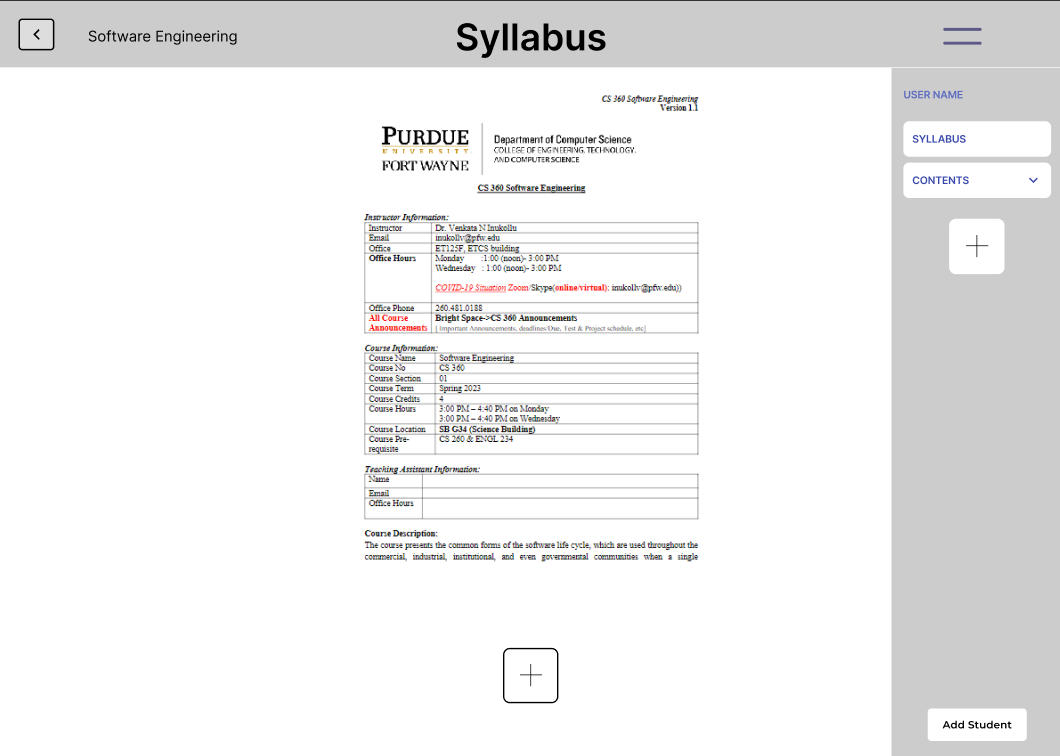
*Category/Subcategory Page*

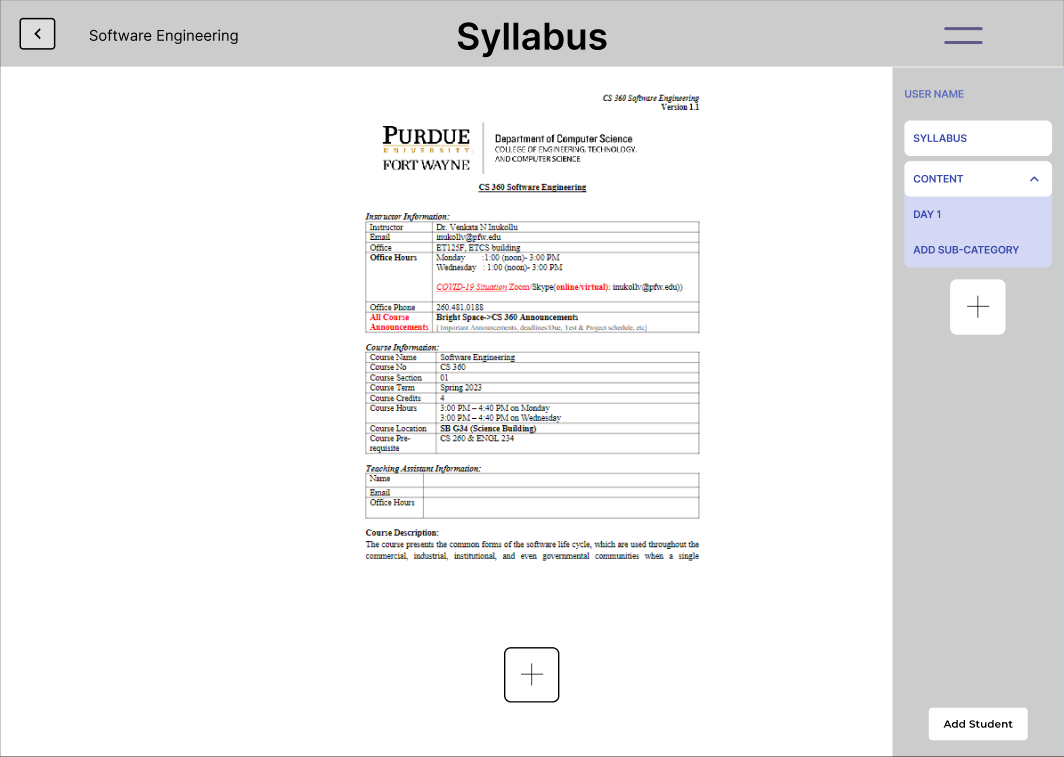
 Back arrow button



A category or subcategory page will display all the components that the teacher has added to the page.

The name of the category/subcategory will be displayed at the top in the center

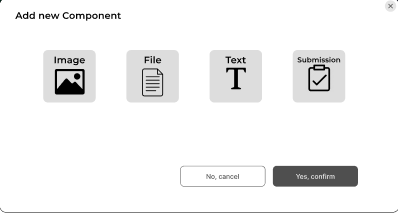
When the category/subcategory name is clicked at the top, if the user is a teacher, then they can edit the name. When off of the text, the changes will be submitted to the database. (Note: This also changes the text in the relevant course/subcourse button on the sidebar)

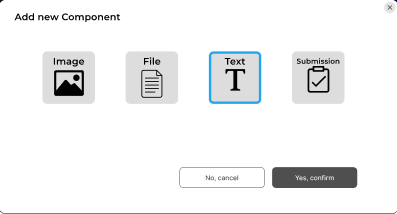
Has add buttons, add person button, add subcategory, sidebar, and categories/subcategories listed in the sidebar, they will act the same way as they did in the course homepage.

The add, add person, and add subcategory buttons will still only be visible to a teacher.

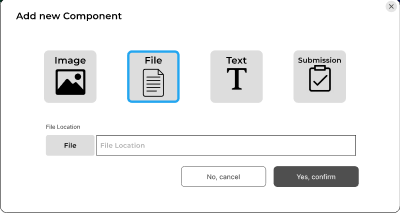
Clicking on the back arrow button on the top left will bring the user back to the homepage.

*Add component Popup*

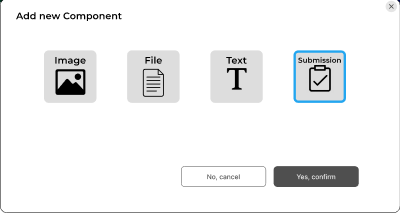


The teacher can specify what type of component they want by selecting the image, file, text, or submission button.

The teacher can change their selection by clicking on a different button.

When image or file are selected, a file location prompt will appear. Clicking on the text field will allow manual input. Clicking the file button will open file explorer when the user can select the file they want to upload.

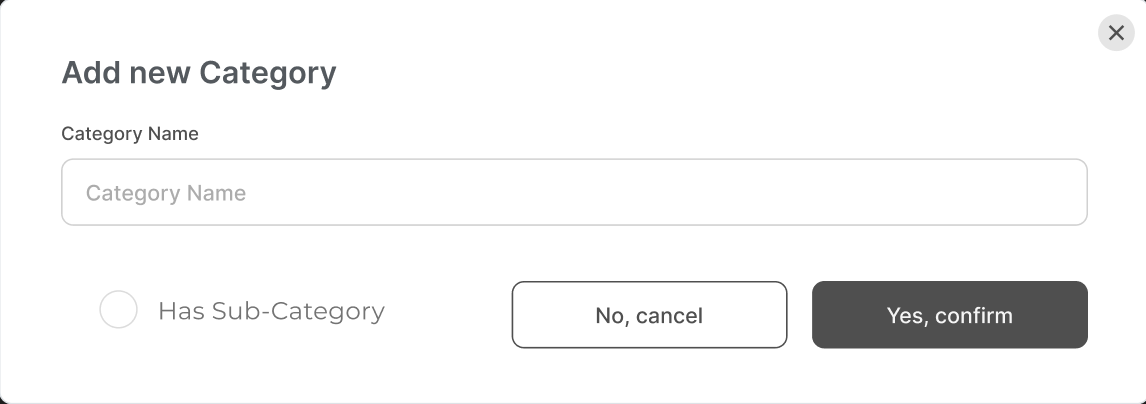
For image selection, the file uploaded must be in an image file type.



In order to submit all necessary fields must be filled out.

Submitting will add the component into the database. The page will then be reloaded to display the added component.

*Add Category/Subcategory Popup*



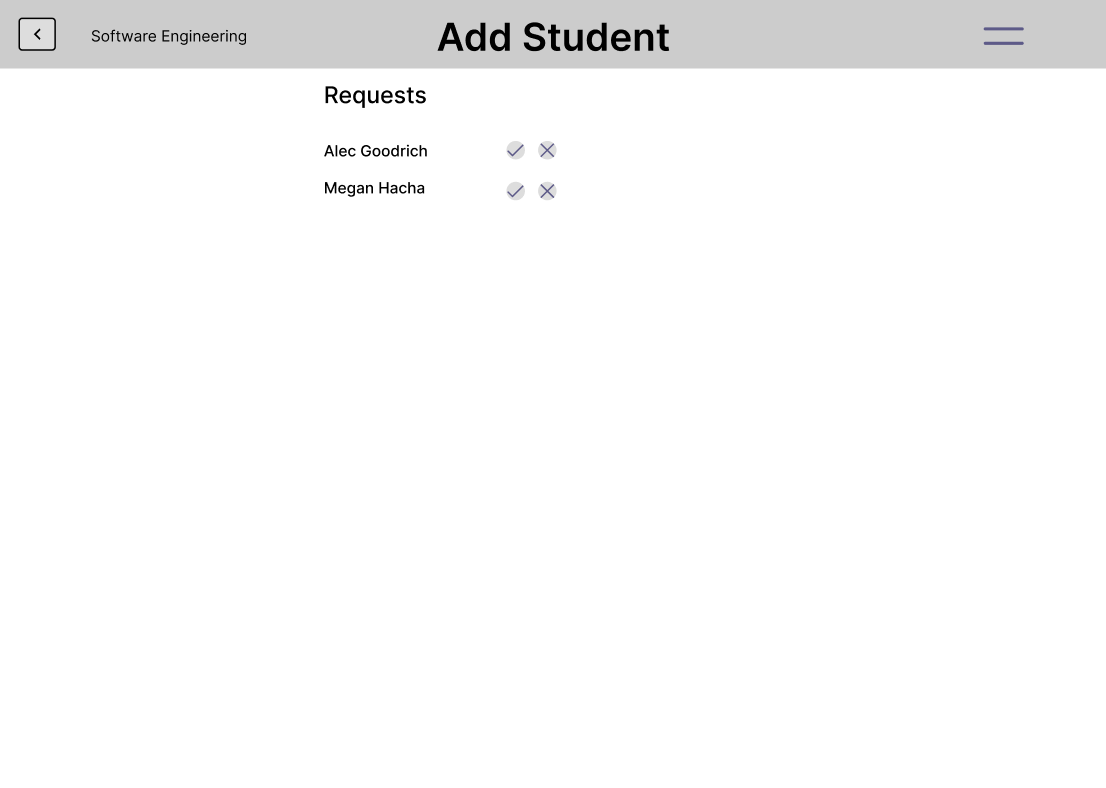
Has fields that will define the category/subcategoy’s name and if it will have subcategories

In order to submit all necessary fields must be filled out.

Submitting will add the category/subcategory into the database. The page will then be reloaded.

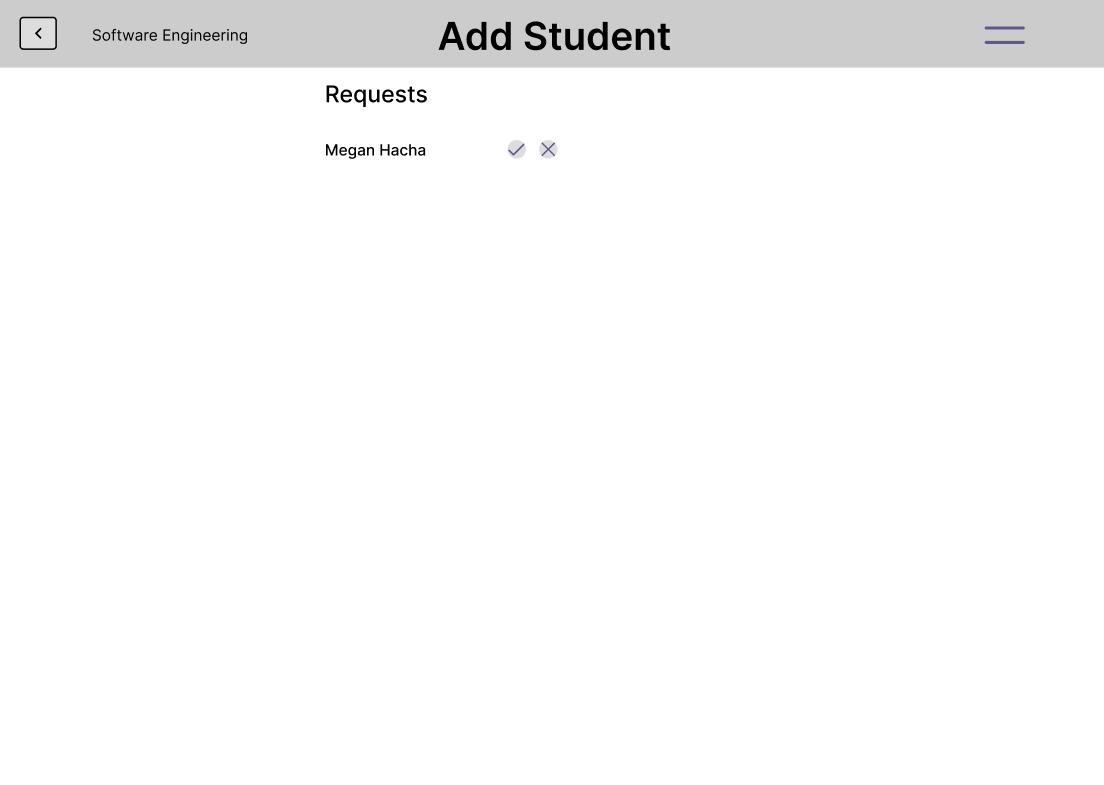
(Note: Subcategories can have subcategories)

*Add Student Page*



Has back, sidebar, back button, and category/subcategory buttons, which will act the same way as in other pages.

The first and last name of students requesting access will be displayed in a list.

Clicking the check mark will allow the student in that row to access the course. A message will be sent to the database to give that student’s account access to the course.

Clicking the x will deny access to the course to the student in that row. A message will be sent to the database to

deny that student’s account access to the course.

*Image Component*



An image component will display the image of the given file for the component.

If the image is too big for the page it will be scaled down to fit the page.

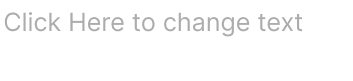
*File Component*



A file component will display the name of the file that was given to the component.

Clicking the file name or the download icon will download the file.

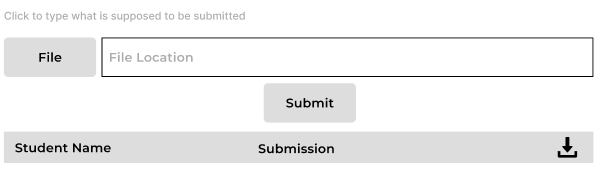
*Text Component*



A text component displays the text that is stored in the component.

Clicking on the component will allow a teacher to type changes to the text. After clicking off, a message with the changes will be sent to the database.

*Submission Component*



To submit a file, the user defines the file location of the file they want to submit.

The user can do this by manually entering in the file location or by clicking the file button. Clicking the file button will bring up file explorer where the user can select a file. To finish the user will click the submit button. This will send a message to the database with the submission.

A teacher will be able to see all of the submissions made by users in a table.

Clicking the download icon next to the submission will download that specific submission.

Clicking the download icon in the table header will download a file containing all submissions.

Clicking on the text field above the file submission area will allow the teacher to type changes When clicked off the changes will be sent to the database.

**Non-Functional Requirements**

Non-functional requirements are used to describe attributes and limits that affect an entire system.

1. **Security**: the system must be made so that the passwords of users are kept safe from theft using encryption.
2. **Usability**: The website will primarily be used on a desktop or laptop computer. In addition, it should be set up in a way that is simple and easy to figure out so it saves confusion for users. This will be done by ensuring a UI that is clear on each function’s purpose and use, along with the site being easy to navigate.
3. **Scalability**: Because the system is used for file uploading and downloading, it must be able to handle a decent amount of traffic for both, along with being able to store large amounts of files. Since this is a site for learners and teachers, it must be able to handle a large number of users without a major effect on performance.
4. **Reliability**: The system must be available a vast majority of time to ensure successful uploading and downloading no matter what time of day. In the event of any critical failures, the system should be returned to normal within ~10 minutes.
5. **Maintainability**: In the event of any components requiring maintenance, fixes must be done within a reasonable timeframe.

**Technical Requirements**

1. GitHub: Software development tool that uses GIT. Used for uploading project files and integrating the different project versions each team member has.
2. SQL: Industry standard programming language to create and manage databases.
3. HTML: Standard markup language for documents designed to be displayed in a web browser.
4. CSS: Stylesheet language that describes the presentation of an HTML document.
5. JavaScript: Standardized object oriented programming language that enables dynamically updating content, multimedia control, and more for a robust webpage.

**Project Risks**

1. Sick team mates and long term absences would affect the amount of work done within a certain time frame. Which could result in incomplete or untested features.
2. Due to limited experience with databases, some features may be incomplete or untested.
3. Unplanned requirement changes or expansion by the client in the final phase will possibly result in inability to complete the new changes by the given deadline.

**Project Management**

| Course Resource Library 1.0 | | | | |
| --- | --- | --- | --- | --- |
| Dates |  |  |  |  |
| Alec | Project Team | Srs Doc/Pres | Design/Sprint Pres | Final Project/Pres |
| Group Project Selection  Project Report Documentation | Prototype Design  Presentation  Client Meeting | Database: File Management  Database: Creation  Presentation | Application Demo  Presentation  Final Documentation |
| Alex | Group Project Selection  Project Report Documentation | User Requirements  Presentation  Client Meeting | Database: Login/ Registration  Database: Creation  Presentation | Application Demo  Presentation  Final Documentation |
| Megan | Group Project Selection  Project Report Documentation | Non-Functional Requirements  Presentation  Client Meeting | UI: Page Components  Presentation | Application Demo  Presentation  Final Documentation |
| Halie | Group Project Selection  Project Report Documentation | Introduction  Presentation  Client Meeting | UI: Page Creation  Presentation | Application Demo  Presentation  Final Documentation |
| Prasanna | Group Project Selection  Project Report Documentation | Technical Requirements  Project Risks  Presentation  Client Meeting | UI: Login/  Registration  UI: Add Student  Presentation | Application Demo  Presentation  Final Documentation |

**Prototype**

Individual Screen States: <https://www.figma.com/file/53MgDaPyagVKghCewbmr5P/Prototype?node-id=0%3A1&t=lOXMrWba1EajDAXn-1>

Interactable: (Not all functions are on every page)

<https://www.figma.com/proto/53MgDaPyagVKghCewbmr5P/Prototype?node-id=4%3A254&scaling=min-zoom&page-id=0%3A1&starting-point-node-id=4%3A254>