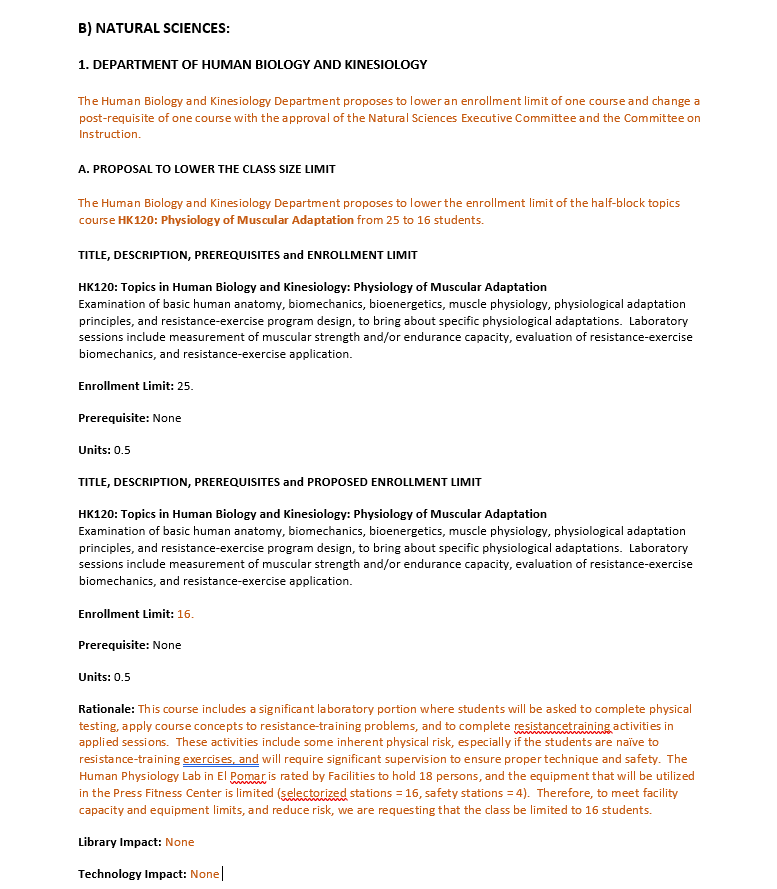
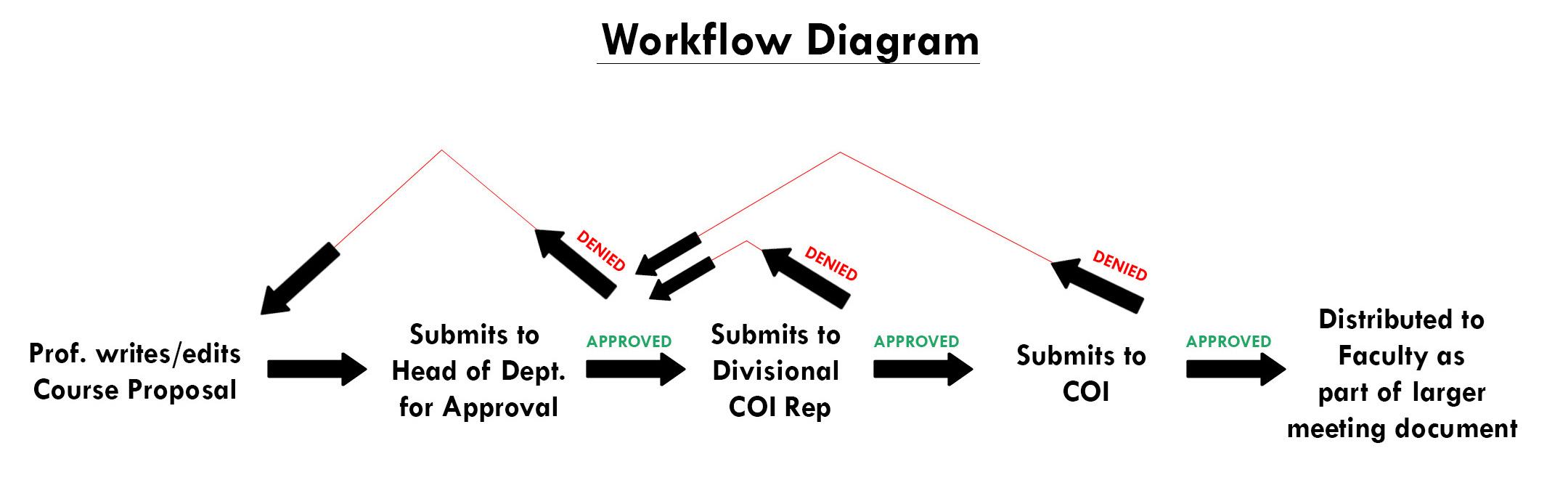
**Introduction:**

Our project aims to streamline the workflow for making course proposals at Colorado College. Currently, professors need to go through a very tedious process, following specific Word document formatting rules and manually retrieving existing course data to add to their proposals, which is time consuming. Our web application is a more efficient way to generate course proposal documents.

Below is an example of such a document. Text which faculty must come up with themselves is colored orange. Black text represents data that is available on the Colorado College course catalog.



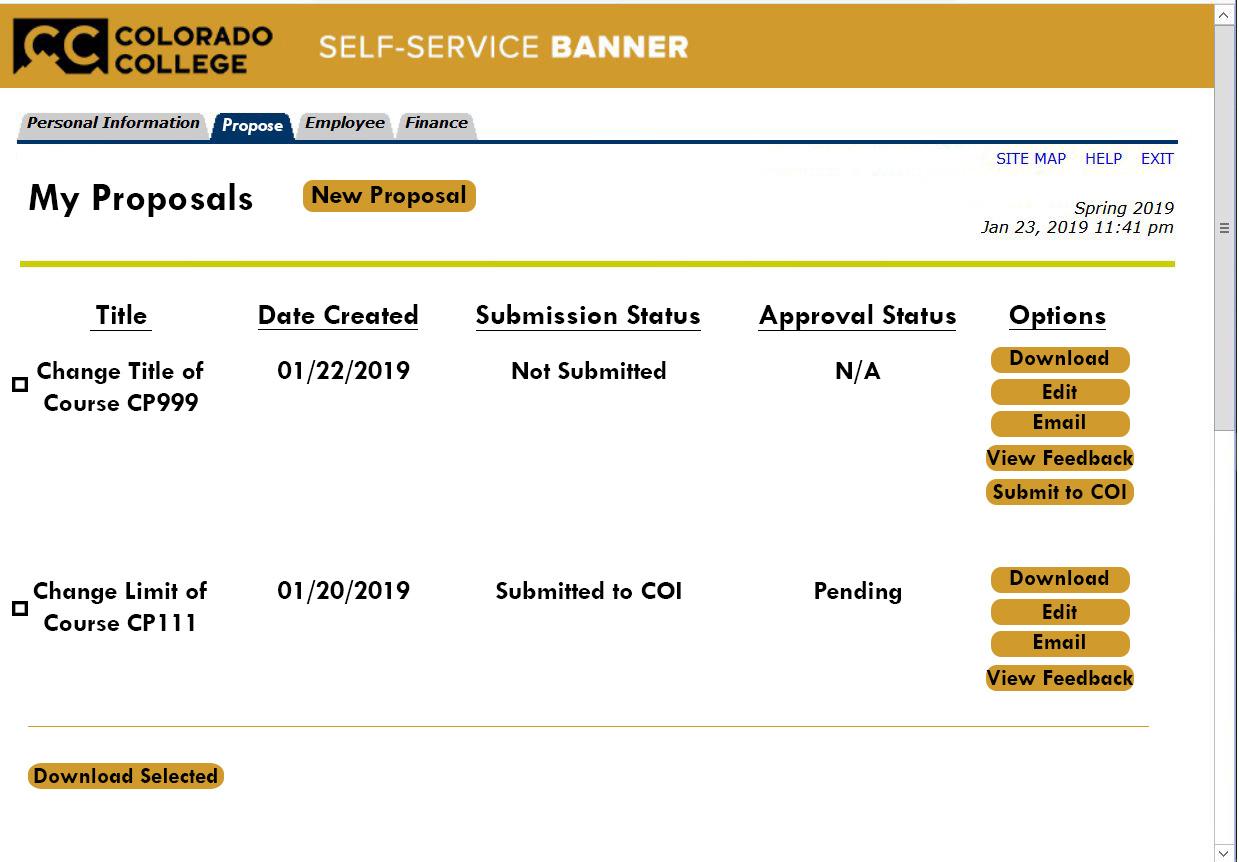
These documents are part of a workflow involving four different authorization levels. Our app aims to get this tedious and confusing process out of the email chain, and into its own integrated system.

****

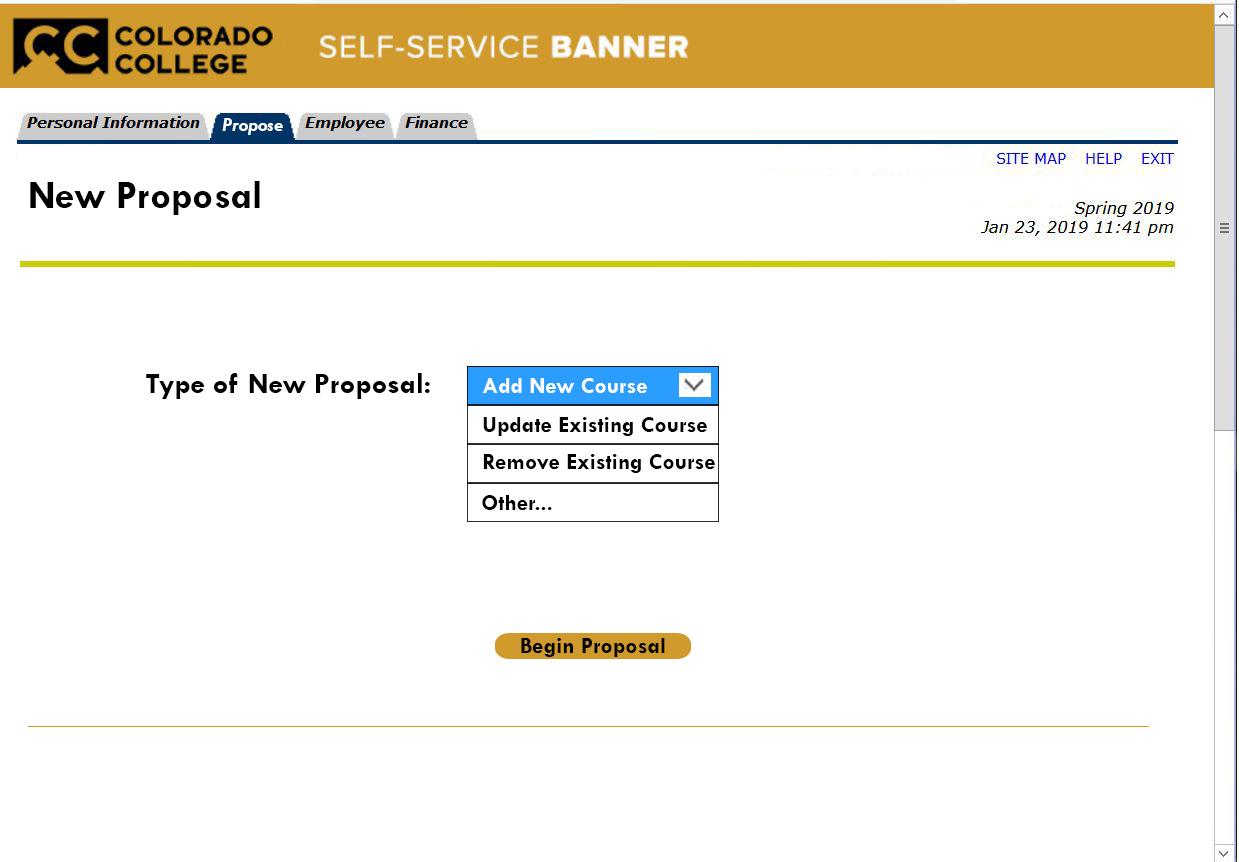
With our application, Professors fill in forms for their desired type of proposal in order to generate perfectly formatted course proposal Word documents. Our application automatically pulls data from a database containing all existing courses’ info, and generates a Word document which users can download and save. The application also supports features of submitting the proposals to different agendas (i.e. head of department, Committee on Instruction) for approval and granting specific privileges to these client groups based on their authentication level. We expect our project to have integration with the CC Single Sign-in system by the end of this project and potential data flow via certain feeds with the Banner system in the future.

**UI**

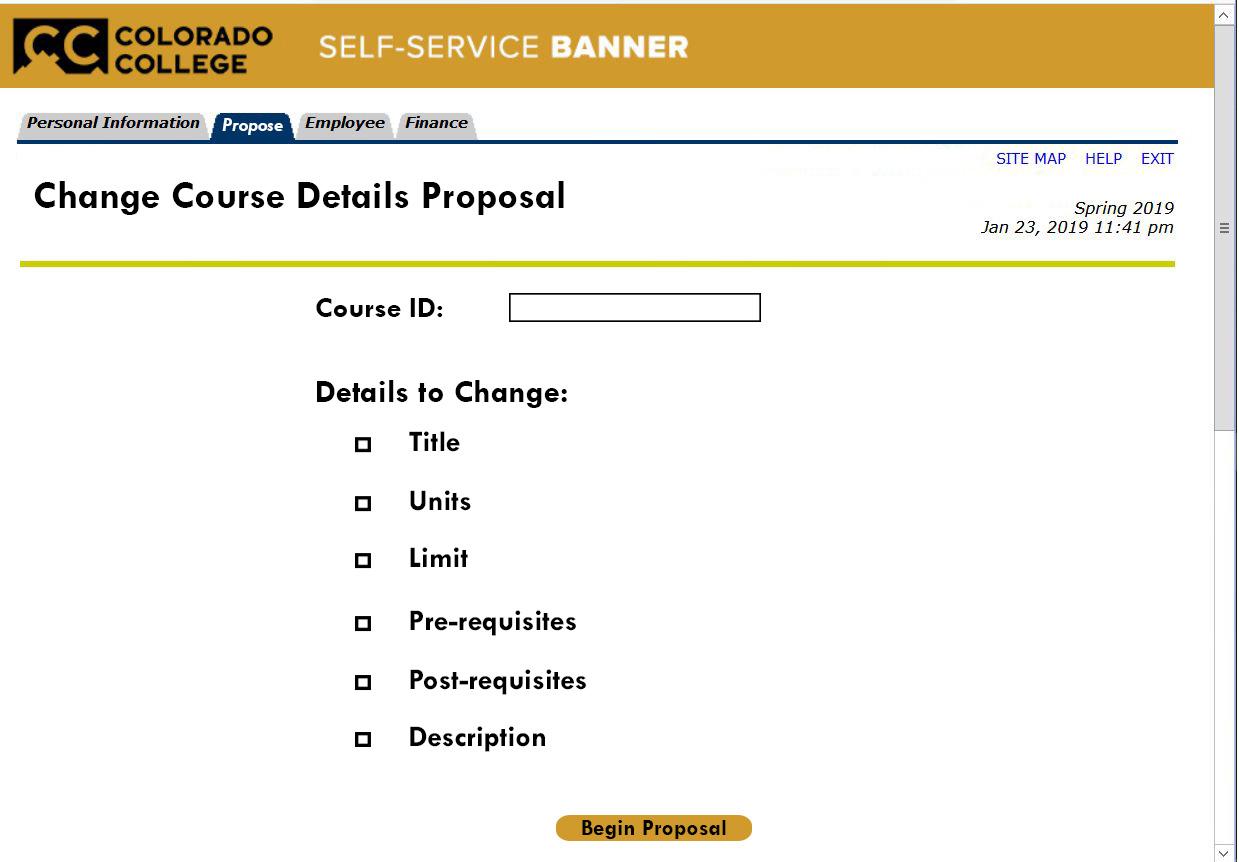
**Home Console - Download, Edit and Submit Proposals**

****

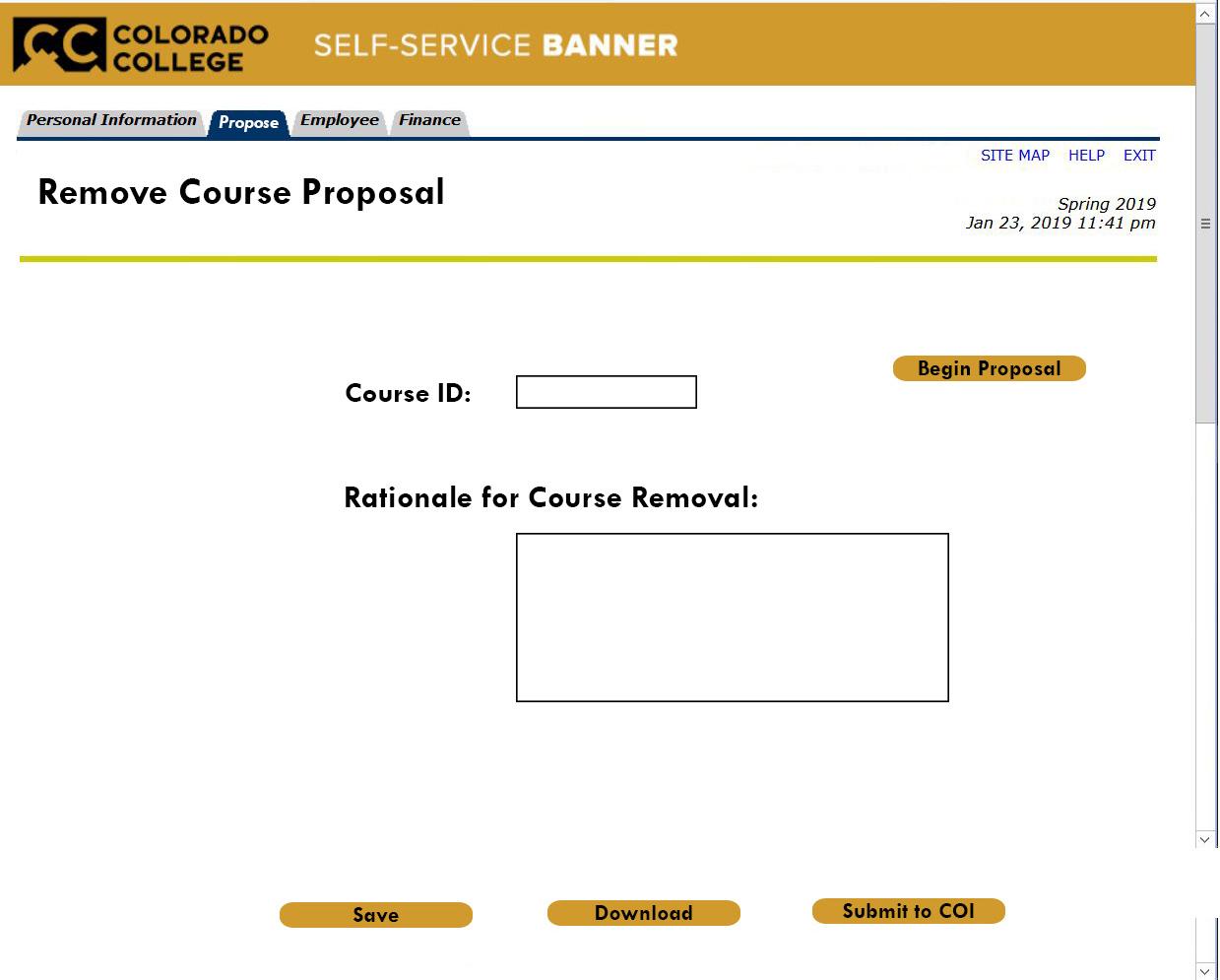
**New Proposal - Choose what kind of proposal to create: Add new course, Update existing course, and Remove existing course**



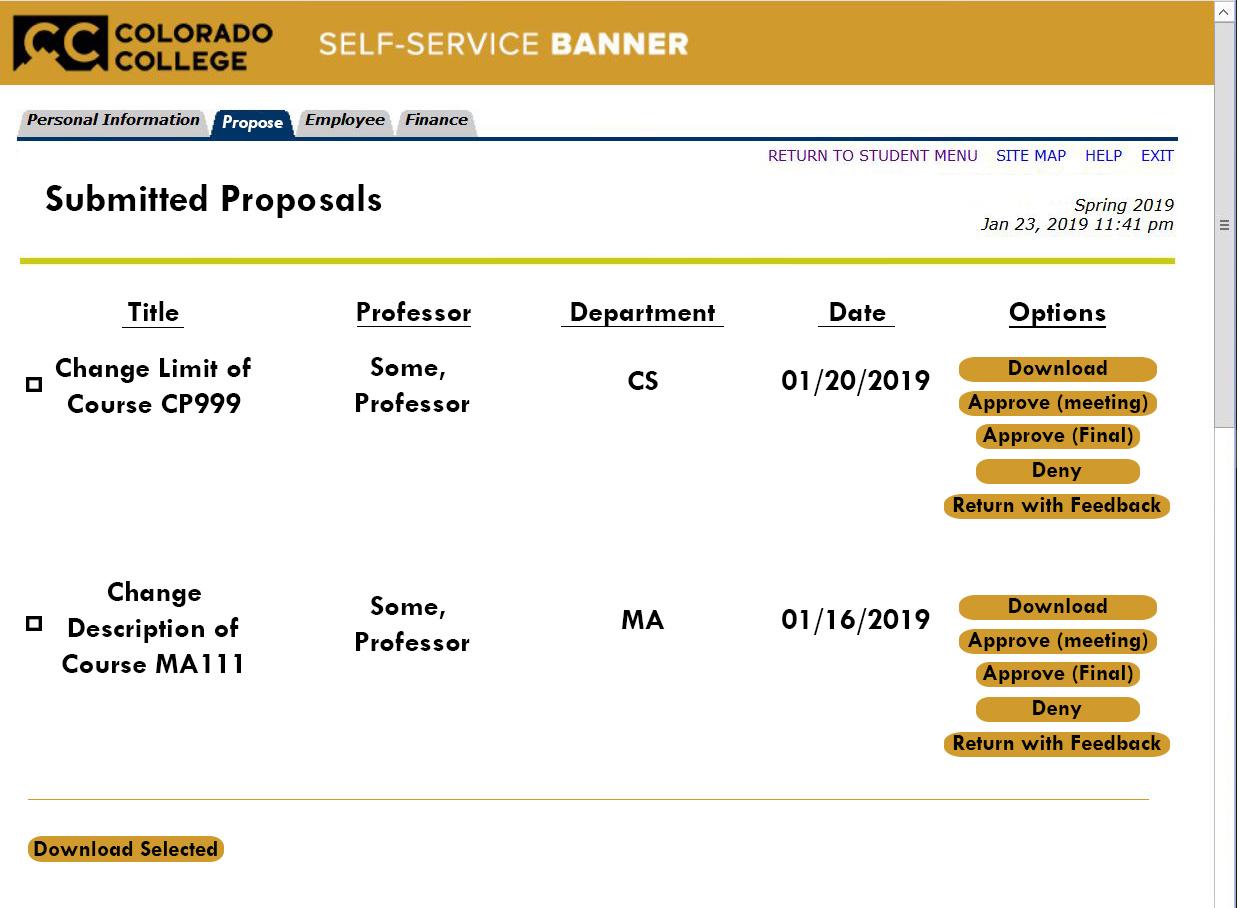
**Update Existing Course Options - Choose what aspects of the course you want to propose changes to.**

****

**Remove Existing Course - Choose which course you are proposing to remove and provide rationale**

****

**Approve/Deny Proposal Console - View, download, approve and deny proposals.**

****

**Users:**

There are four types of users on our platform: **Professors**, **Department Heads**, **Divisional Committee on Instruction reps**, and **Committee on Instruction members**. Each user type has different needs, which are outlined in the **Progress Summary** section. These groups are separated by their permission level: professors have permission level 1, department heads have permission level 2, divisional Committee on Instruction (a.k.a. COI) reps have permission level 3, and COI members have permission level 4.

All four user types have the ability to create proposals. When a Faculty member creates a proposal, it is first submitted to the **Department Head** for their department for approval. If the **Department Head** denies the proposal, it is returned to the Faculty member; if the proposal is approved, it is passed on to the **Divisional COI rep** for the division that the department falls under. If the **Divisional COI rep** denies the proposal, it is returned to the **Department Head** who submitted the proposal; if the proposal is approved, it is passed on to the **COI members**. At this point, the proposal is either approved to go to the Committee on Instruction meeting agenda for that block, or it is denied and returned to the **Divisional COI rep** who submitted it.

**Specification Traceability**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| User Story # | User Story Desc | Implementation | Architecture Component | Test Case Description |
| 1 | Log In to Web App | Log-in page and form | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can enter username and password for Test user to successfully be redirected to /home |
| 2 | Choose New Proposal Type | Drop-down menu | APACHE, HTML, CSS + PHP | Test if Test user can select each different proposal type (arrive at appropriate page after clicking “Begin Proposal”) |
| 3 | Create Add a New Course proposal | Navigate to Add a New Course proposal webpage + form | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can open, fill out and submit Add New Course Proposal form, which is added to DB |
| 4 | Create change Existing Course proposal ( and change all possible Course criteria) | Navigate to Change an Existing Course webpage + form | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can open, fill out and submit Change Existing Course proposal form, which is added to DB |
| 5 | Create remove Existing Course proposal | Navigate to Remove an Existing Course webpage + form | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can open, fill out and submit Remove an Existing Course proposal form, which is added to DB |
| 6 | Create Change Major proposal | Navigate to Change a Major webpage + form | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can open, fill out and submit Change Major proposal form, which is added to DB |
| 7 | Create Change Minor proposal | Navigate to Change a Minor webpage + form | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can open, fill out and submit Change Minor proposal form, which is added to DB |
| 8 | Create Custom Proposal | Navigate to Create Custom Proposal webpage + form | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can open, fill out and submit Customized proposal form, which is added to DB |
| 9 | Download a Word doc of proposal | Download button on Home page | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can download a Test proposal with the expected output |
| 10 | Auto-fill Word doc with existing course info (when relevant) | Database Querying | APACHE, PHP + MySQL | Test if Test user's downloaded Test proposal includes information not included in proposal |
| 11 | Submit Proposal to Head of Dept | Click “Submit” button | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can click SUBMIT button for proposal on Home pagel to change submission status value |
| 12 | Check Proposal Submission Status | Home page | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can see Test proposal's submission status from Home page |
| 13 | Check Proposal Approval Status | Home page | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can see Test proposal's approval status from Home page |
| 14 | Edit a Proposal | Click “edit” button | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can click EDIT button on proposal to edit the existing proposal's fields by being redirected to the proper webpage + form |
| 15 | Send Proposal to faculty for feedback | Click “Get Feedback” button | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can click GET FEEDBACK button and send proposal to desired users by entering desired faculty member’s names |
| 16 | View Proposal's Feedback | Click “View Feedback” button | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can click VIEW FEEDBACK button and view all of proposal's feedback from desired faculty |
| 17 | Give Feedback on a received Proposal | ----- | APACHE, HTML, CSS, PHP, MySQL | Yet to implement |
| 18 | Download Series of Selected Proposals | Select multiple proposals via checkboxes and click “Download All” at bottom of page | APACHE, HTML, CSS, PHP, MySQL | Test if Test user can check multiple Test proposal boxes and click DOWNLOAD ALL to download one compiled Test Word doc in the expected output |
| 19 | Easily Rebuild System via documentation | Appropriate documentation |  | Include Documentation necessary for anyone to rebuild our system from scratch |
| 20 | Understand Source Code | Appropriate documentation |  | Include Documentation necessary for anyone to maintain/edit the code for our system if need be |

**Testing & Validation:**

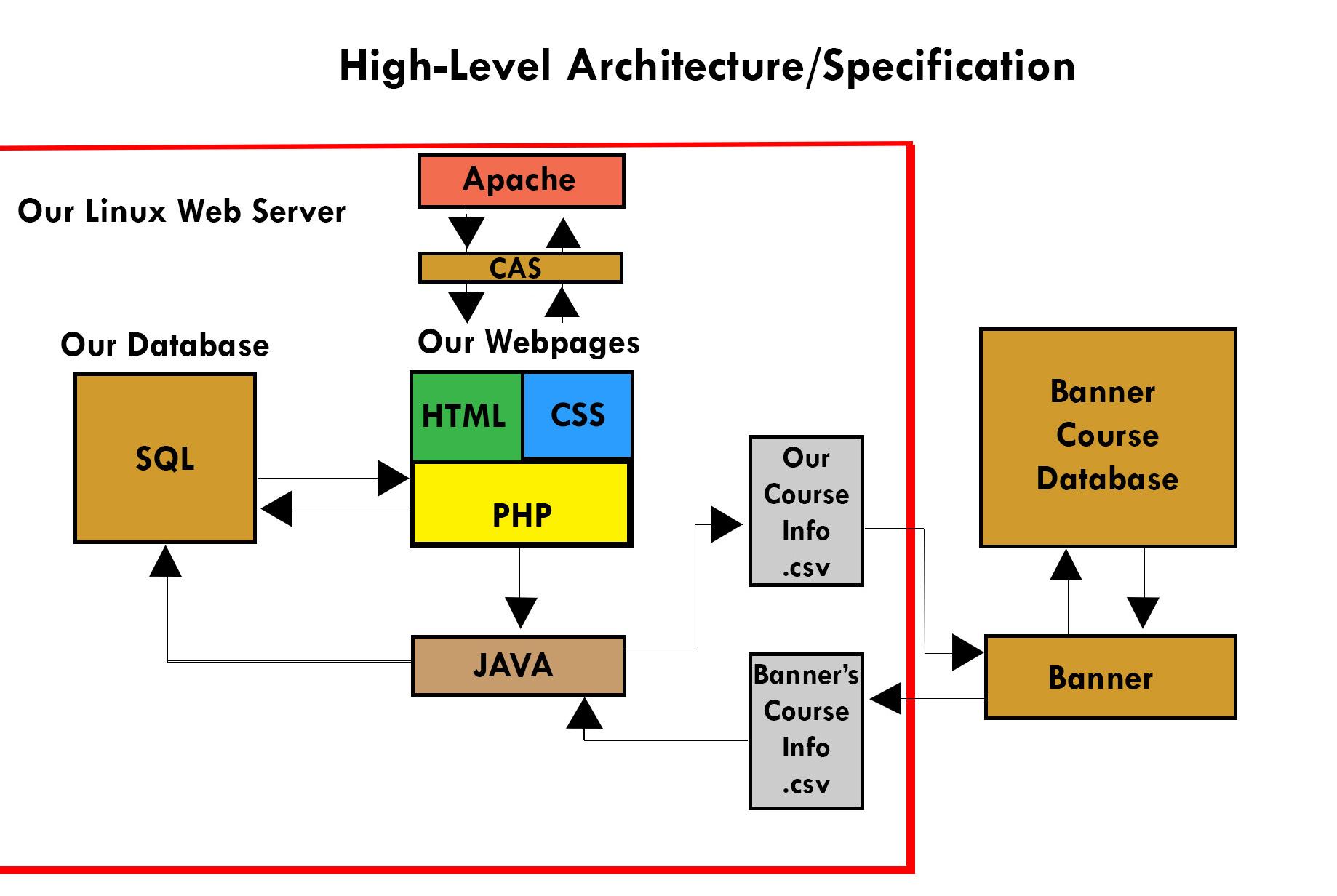
Our web application is just that - a web-based application that functions primarily through the use of HTML forms and buttons to perform specific PHP functions, such as submit an “Add a New Course” proposal or download a Word doc version of a proposal. In order to effectively test the capabilities of our web-app, we have to be able to programmatically fill out specific web-page forms (as well as query our database to ensure the form queried our database properly), click buttons (as well as check the URL reached after clicking), and read content generated on each page (to confirm that our web-pages can query the database correctly to display info).

The best way that we would be able to achieve this would be to create a Test user program, with very specific tests that the Test user could run to perform functions such as create an “Add a New Course” proposal via our web-form, and then query the database for the entered proposal to see if the output is as desired. For non-form related functionality, the Test user could go to a specific page (such as the Home page) knowing that certain data should be displayed, and then parse through the HTML on the page until it found the desired data; if any data were not found, then the test would fail, but otherwise it would pass. Such a Test program would also need to be able to confirm that buttons sent them to the correct links when clicked, as well as performed the correct functions such as download a Word doc. Finally, we will need to implement some sort of functionality to this Test user that will allow them to create different kinds of proposals and then determine whether or not the generated Word document for each proposal was created with the desired formatting rules and required information.

After doing some research on industry-standard testing of web-based applications, we found Cypress, an open-source, front-end testing tool. Cypress is able to simulate commands such as entering text into a form and clicking buttons on web-pages. While we are still reading up on the documentation included with Cypress, it is looking as though it will be a very useful tool in testing the functionality of our web-app.

**Design Documentation:**

Our web-app utilizes PHP in order to communicate with an SQL database (DB), which contains the details for all Courses. Users (information of which is stored in our DB) create proposals, which are stored in the Proposals table in our DB. When a proposal is submitted to the Head of the Department/COI for review, it is either accepted, denied, or returned with feedback. If the proposal is accepted all the way through the hierarchy, the proposal is added to the Committee on Instruction’s meeting agenda for the upcoming block.

****

**Component Overview:**

**External Components:**

**Banner Course Database:**

A database which holds records for all courses at CC.

**Banner:**

The point of interface between our application data and the **Banner Course Database**

**Internal Components:**

**Linux Server:**

A linux machine running CentOS 7.6

**Apache:**

Used for web hosting. Handles https requests and serves the browser of the client

**CAS:**

**CAS** is Colorado college’s single sign in authentication system. Our web server checks for an authentication token, which is produced by **CAS**. If this token is not present or invalid, the user is routed to cas.coloradocollege.edu. There, they are prompted to provide their CC username

and password. If their credentials are good, an access token is produced and they are redirected once again to our domain (proposal-tool.coloradocollege.edu)

**Webpages:**

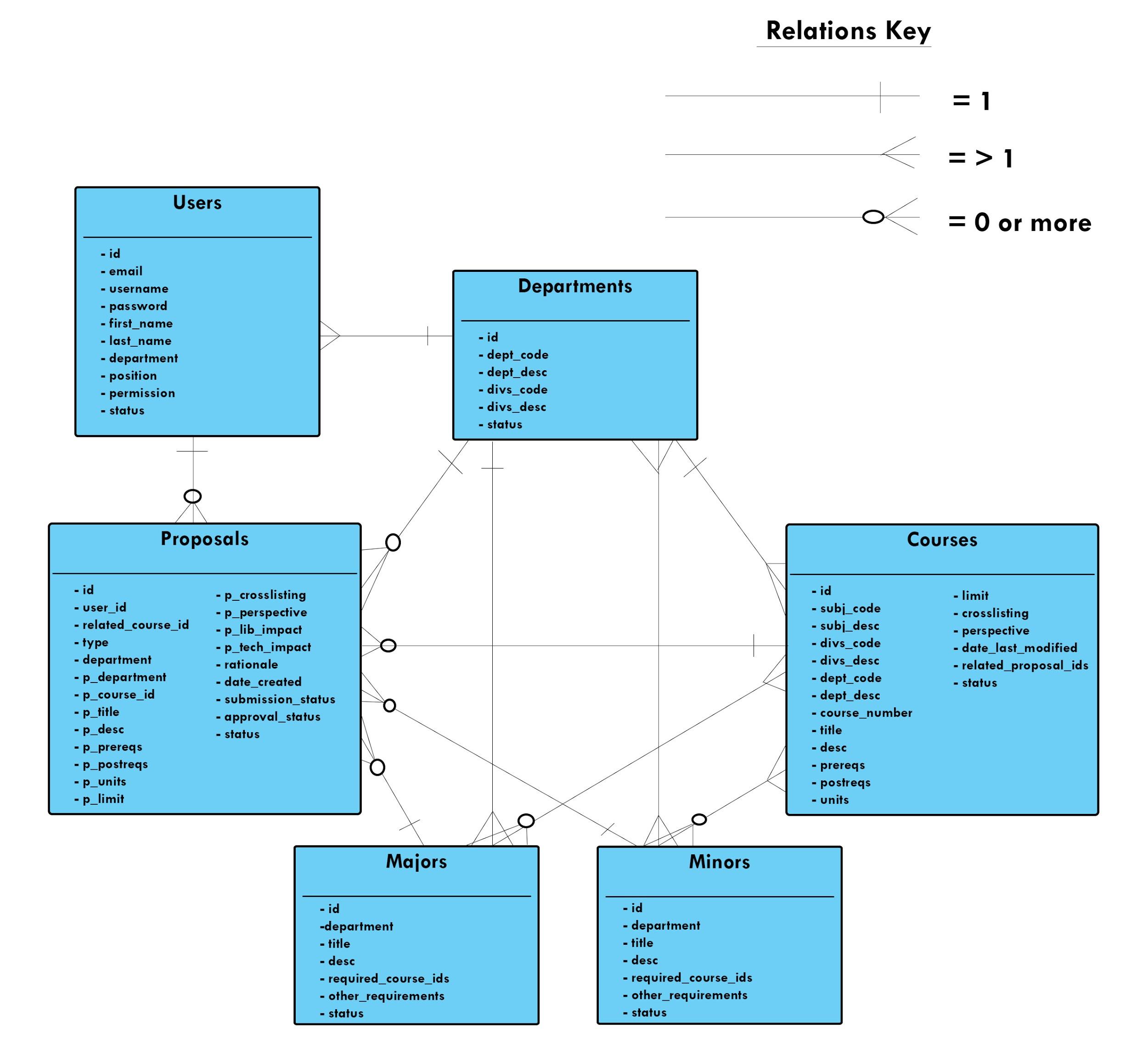
Our webpages use **HTML** and **CSS** for front end UI. Backend functionality for the webpages is written in **PHP.** **PHP** also queries our **local SQL database** to fill in relevant information in course proposal web forms, and course proposal documents.

**Java:**

**Java** is used to continually update our server. Once a day, **Banner** sends our application their course catalog, formatted as a.csv file. **Java** is used to extract this data and enter it into our **local SQL database**. In addition, **Java** is also used to produce a .csv file containing all accepted changed, which is sent to **Banner.**

**Local SQL database:**

**Database Entity Relationship Diagrams**

****

**Hardware requirements:**

Server: 1-Core X86 CPU, 1GB RAM, 25 GB storage, Network connection

Client: Network connection, Can run web browser, 500MB storage

**Progress Summary**

This week, our team faced a number of logistical challenges. Per the suggestion of ITS, we decided to explore using a third party host for our web server. We deployed our app to a LAMP server hosted on DigitalOcean. On Wednesday, we sat down with members of ITS who informed us that they could in fact give us an on campus web server to deploy our app. Furthermore, we are allowed access to the cas subdomain, meaning users will be able to sign in with their CC username and password. This was great news for us.

On the development end, we completed front and back end functionality for two core user stories central to our applications functionality. With our web app, users may now fill a web form to create a proposal document to add a new course, or to change an existing course. Users may also download these documents via the Download button on the Home page.

We updated our data model to include a table for departments, majors, and minors. In addition, we added new fields to our existing tables to account for new user stories.

Below is our list of user stories and who is responsible for completing them:

**Completed High Priority Low Priority IceBox**

**[Log In]**

As a professor, I want to be able to log in to the web app in order to use the web app.

**[Add a New Course]**

As a professor, I want to be able to propose a new course in order to add it to the course catalog/schedule for registration.

**[Change Course Department]**

As a professor, I want to be able to change a course’s department in the course catalog/schedule in order to make the related department make more sense for the specific course.

**[Change Course ID]**

As a professor, I want to be able to change a course ID in the course catalog/schedule in order to make the course ID more appropriate for the course.

**[Change Course Title]**

As a professor, I want to be able to change a course title in the course catalog/schedule in order to make the title more accurate.

**[Change Course Description]**

As a professor, I want to be able to change a course description in the course catalog/schedule in order to make the course details always relevant.

**[Change Course Unit]**

As a professor, I want to be able to change the unit of a course in order to have proper units set up for various formats of class.

**[Change Course Prerequisite]**

As a professor, I want to be able to change the prerequisites for a certain course in order to make sure students get prepared to take this class.

**[Choose Proposal Type]**

As a professor, I want to be able to choose the type of proposal by a pull-down menu in order that the form can display the appropriate fields to fill in.

**[Pull Out Course Data]**

As a professor, I want the web form to retrieve the data of a course automatically so that I do not have to search for the data and type it in manually.

**[Save Proposal as a Copy]**

As a professor, I want to be able to download my proposal in order to save a copy of it.

**Jia**

**[Remove an Existing Course]**

As a professor, I want to be able to remove a course from the course catalog/schedule in order to keep students from registering an unoffered course.

**Christian**

**[Submit my Proposal]**

As a professor, I want to be able to submit my proposal to the administrative level in order to get their approval.

**Harrison**

**[Edit my Proposal]**

As a professor, I want to be able to edit an in-progress proposal in order to keep it updated until finished.

**Christian**

**[View Submitted Proposals]**

As the head of the Department, I want to be able to view the submitted proposals with dates, submitters and departments in order to perform further actions on the proposal.

**Christian**

**[Save Submitted Proposal]**

As the head of the Department, I want to be able to download the submitted proposal in order to keep a record of it.

**Jia**

**[Approve Proposals]**

As the head of the Department, I want to be able to approve a proposal that was submitted to me in order to have it for further discussion by members of the COI.

**Christian**

**[View Submitted Proposals]**

As a member of COI, I want to be able to view the submitted proposals with dates, submitters and departments in order to perform further actions on the proposal.

**Christian**

**[Save Submitted Proposal]**

As a member of COI, I want to be able to download the submitted proposal in order to keep a record of it.

**Christian**

**[Check Proposal Submission Status]**

As a professor, I want to be able to check the submission status for my proposals in order to see if it has been submitted to the Committee of Instruction.

**Harrison**

**[Customize a Proposal]**

As a professor, I want to be able to customize a type of proposal by having free combination of different course attributes in order to propose something not included in the category.

**Jia**

**[Check Proposal Approval Status]**

As a professor, I want to be able to check the approval status for my proposals in order to see if my submitted proposal are under reviewing, approved or denied.

**Jia**

**[Send Proposal for Feedback]**

As a professor, I want to be able to send out my proposals to other professor colleagues in order to get feedback from them.

**Jia**

**[View Proposal Feedback]**

As a professor, I want to be able to view the proposal feedback returned by other professor colleagues in order to revise or finalize the proposal for submission.

**Jia**

**[Give Feedback on Proposal Received]**

As a professor, I want to be able to view the proposal feedback returned by other professor colleagues in order to revise or finalize the proposal for submission.

**Harrison**

**[Save a Series of Selected Proposals]**

As a professor, I want to be able to download all selected proposals in order to save a copy of several proposals as a single document.

**Christian**

**[Check Course Proposal History]**

As a professor, I want to be able to view a report of historical proposals related with a certain course in order to get an idea of the chronicle changes of the course.

**Harrison**

**[Save a Series of Selected Submitted Proposals]**

As the head of the Department, I want to be able to download a couple of selected, submitted proposals in order to keep a copy of multiple proposals.

**Jia**

**[Deny a Proposal]**

As the head of the Department, I want to be able to deny a proposal in order to have the submitter reconsider the proposal or make some radical changes.

**Jia**

**[Return a Proposal with Feedback]**

As the head of the Department, I want to be able to return a proposal with feedback in order to have the submitter make some minor changes and re-submit it later.

**Harrison**

**[Save a Series of Selected Submitted Proposals]**

As a member of COI, I want to be able to download a couple of selected, submitted proposals in order to keep a copy of multiple proposals.

**Jia**

**[Approve Proposals for Meeting]**

As a member of COI, I want to be able to approve a proposal to go to the meeting in order to have it for further discussion by other committee members.

**Jia**

**[Approve Proposals for Finalization]**

As a member of COI, I want to be able to approve a proposal to finalize it in order to put the modified course in the catalog/schedule.

**Jia**

**[Deny a Proposal]**

As a member of COI, I want to be able to deny a proposal in order to have the submitter reconsider the proposal or make some radical changes.

**Jia**

**[Return a Proposal with Feedback]**

As a member of COI, I want to be able to return a proposal with feedback in order to have the submitter make some minor changes and re-submit it later.

**Jia, Harrison and Christian**

**[Share Data with The Registrar]**

As a member of the Registrar, I want to be able to easily enter this data into the course catalog so that the course catalog is updated.

**Jia, Harrison and Christian**

**[Easily Rebuild]**

As a software developer, I want to be able to easily rebuild the entire system so that I can fix it when the entire system breaks.

**Jia, Harrison and Christian**

**[Understand Source Code]**

As a software developer, I want to be able to easily understand source code so that I can more easily add new features and debug.