

Table of Contents:

[1. User Guide \(pg. 3\)](#)

[1.1 Introduction \(pgs. 3 - 5\)](#)

[1.2 Building the COI Agenda \(pgs. 5 - 9\)](#)

[1.2.1 Level 4 - Complete COI Agenda](#)

[1.2.2 Level 3 - Division Compiled Department Proposals](#)

[1.2.3 Level 2 - Compiled Department Proposals](#)

[1.2.4 Level 1 - Proposal](#)

[1.3 Proposal Types \(pgs. 9 - 14\)](#)

[1.3.1 Add a New Course:](#)

[1.3.2 Revise an Existing Course:](#)

[1.3.3 Drop an Existing Course:](#)

[1.3.4 Major and Minors:](#)

[1.3.5 Revise a Major or Minor:](#)

[1.3.6 Add a New Thematic Minor:](#)

[1.4 Create a Web Proposal Walkthrough \(pgs. 15 - 21\)](#)

[1.4.1 Add a New Course Proposal](#)

[1.4.2 Change an Existing Course Proposal](#)

[1.4.3 Remove an Existing Course Proposal](#)

[1.4.4 Edit a New Course Proposal](#)

[1.4.5 Edit a Change Existing Course Proposal](#)

[1.4.6 Edit a Remove Existing Course Proposal](#)

[1.5 Current Feature & Further Implementation \(pgs. 22 - 25\)](#)

[1.5.1 Our Current Feature](#)

[1.5.2 What is Coming Next](#)

[2. Developer Guide \(pg. 26\)](#)

[2.1 User Stories \(pgs. 26 - 31\)](#)

[2.1.1 Completed](#)

[2.1.2 In Progress:](#)

[2.1.3 Icebox](#)

[2.2 System Architecture \(pgs. 31 - 33\)](#)

[2.2.1 External Components](#)

[2.2.2 Internal Components:](#)

[2.3 File System Description \(pgs. 33 - 35\)](#)

[2.4 Validation \(pgs. 36 - 37\)](#)

[2.4.1 Back-End Testing](#)

[2.4.2 Front-End Testing](#)

[3. Deployment Guide \(pg. 38\)](#)

[3.1 Local Development Environment \(pgs. 38 - 41\)](#)

[3.2 Deployment \(pgs. 41 - 42\)](#)

[3.3 Maintenance \(pgs. 42 - 43\)](#)

[3.3.1 Data stream](#)

[3.3.2 Software updates](#)

[APPENDIX A \(pg. 44\)](#)

[A.1 Add a New Course \(pgs. 44 - 45\)](#)

[A.2 Revise an Existing course \(pgs. 45 - 46\)](#)

[A.3 Drop an Existing Course \(pg. 46\)](#)

[A.4 Revise a Major/Minor \(pgs. 47 - 48\)](#)

1. User Guide

1.1 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.

Team Software Project - Streamline Course Proposals
Final Demo Documentation
Christian Kennedy, Harrison Selle, Jia Kang

B) NATURAL SCIENCES:

1. DEPARTMENT OF HUMAN BIOLOGY AND KINESIOLOGY

The Human Biology and Kinesiology Department proposes to lower an enrollment limit of one course and change a post-requisite of one course with the approval of the Natural Sciences Executive Committee and the Committee on Instruction.

A. PROPOSAL TO LOWER THE CLASS SIZE LIMIT

The Human Biology and Kinesiology Department proposes to lower the enrollment limit of the half-block topics course **HK120: Physiology of Muscular Adaptation** from 25 to 16 students.

TITLE, DESCRIPTION, PREREQUISITES and ENROLLMENT LIMIT

HK120: Topics in Human Biology and Kinesiology: Physiology of Muscular Adaptation
Examination of basic human anatomy, biomechanics, bioenergetics, muscle physiology, physiological adaptation principles, and resistance-exercise program design, to bring about specific physiological adaptations. Laboratory sessions include measurement of muscular strength and/or endurance capacity, evaluation of resistance-exercise biomechanics, and resistance-exercise application.

Enrollment Limit: 25.

Prerequisite: None

Units: 0.5

TITLE, DESCRIPTION, PREREQUISITES and PROPOSED ENROLLMENT LIMIT

HK120: Topics in Human Biology and Kinesiology: Physiology of Muscular Adaptation
Examination of basic human anatomy, biomechanics, bioenergetics, muscle physiology, physiological adaptation principles, and resistance-exercise program design, to bring about specific physiological adaptations. Laboratory sessions include measurement of muscular strength and/or endurance capacity, evaluation of resistance-exercise biomechanics, and resistance-exercise application.

Enrollment Limit: 16.

Prerequisite: None

Units: 0.5

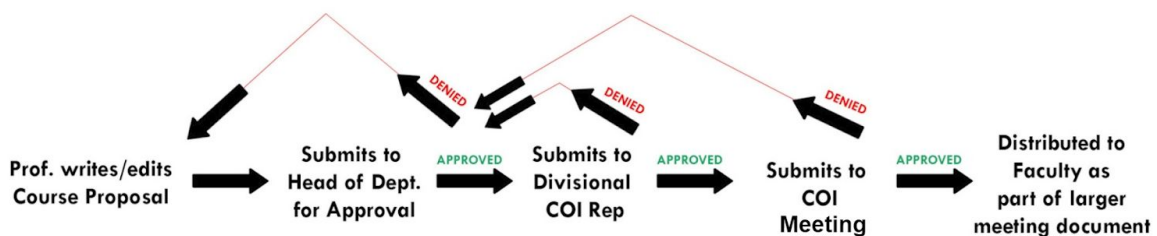
Rationale: This course includes a significant laboratory portion where students will be asked to complete physical testing, apply course concepts to resistance-training problems, and to complete ~~resistance training~~ activities in applied sessions. These activities include some inherent physical risk, especially if the students are naive to resistance-training ~~exercises, and~~ will require significant supervision to ensure proper technique and safety. The Human Physiology Lab in El Pomar is rated by Facilities to hold 18 persons, and the equipment that will be utilized in the Press Fitness Center is limited (~~selectorized~~ stations = 16, safety stations = 4). Therefore, to meet facility capacity and equipment limits, and reduce risk, we are requesting that the class be limited to 16 students.

Library Impact: None

Technology Impact: None

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

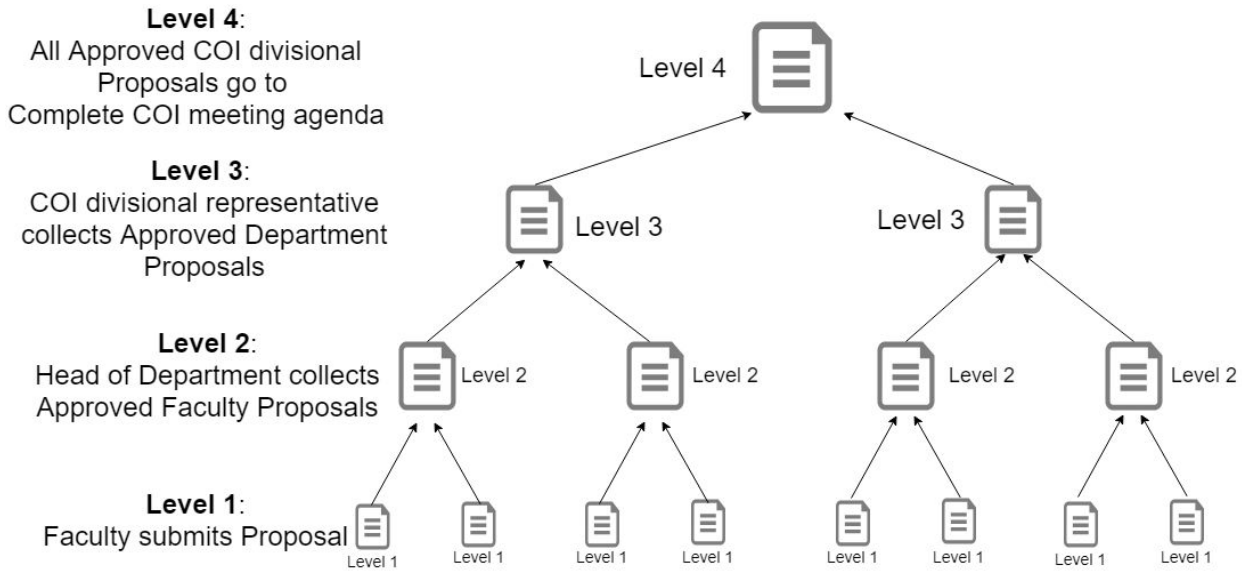
Workflow Diagram



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' information, 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.

1.2 Building the COI Agenda

This section contains information on how the **COI agenda** is generated. The COI agenda is a compilation of all proposals, organized by department and by division. These <> indicate a placeholder for information which must be generated on a case-by-case basis (ie. it is not the same for every COI Agenda) **Orange text** indicates a placeholder for information that is compiled lower down in the tree.



1.2.1 Level 4 - Complete COI Agenda

Table of Contents:

Colorado College Memorandum

To: Committee on Instruction

- **Administration:** <COI administrators, currently, these people are Sandra Wong (Co-Chair), Pedro de Araujo (Co-Chair), Karen Obrzut (Secretary)>
- **Faculty Representatives:** <Divisional representatives, currently Jonathan Lee (Humanities), Victoria Levine (Interdisciplinary Programs), Murphy Brasuel (Natural Sciences), Jim Parco (Social Sciences)>
- **Staff:** <Additional staff, currently Phil Apodaca (Registrar, ex officio), JoAnn Jacoby (Library Director)>
- **Students:** <Students> (unclear what these peoples roles are)

From: <Dean of faculty, currently Sandra Wong>

Date: <Date>Wednesday, January 30, 2019

<Block X> **COI AGENDA** Committee on Instruction <Date>, <Time> <Location>

OVERVIEW

A) HUMANITIES: (Item A. 1-3)

1. <Humanities department with proposals ie. Department of English.> 2. <Second Humanities department with proposed changes> 3.<etc...>

B) NATURAL SCIENCES: (Item B. 1-3)

1. <Natural Sciences department with proposals ie. Department of Mathematics and Computer Science.> 2. <Second Natural Sciences department with proposed changes> 3. <etc...>

C) SOCIAL SCIENCES: (Item C. 1-3)

1. <Social Sciences department with proposals ie. Department of Anthropology.> 2. <Second Social Sciences department with proposed changes> 3. <etc...>

D) INTERDISCIPLINARY PROGRAMS: (Item D. 1-3)

1. <Interdisciplinary programs with proposals ie. Program of Race, Ethnicity, and Migration studies..> 2. <Second Interdisciplinary program with proposed changes> 3. <etc....>

E) GENERAL STUDIES: (Item E. 1-2)

Note: These are proposals that do not fall under a department or division. Such a proposal can include half-block courses and proposals for new Thematic Minors.

1. <Proposal group 1. ie “The Colket Center”> 2. <Proposal group #2> 3. <etc..>

F) CRITICAL PERSPECTIVES COURSES: (Items F. 1-8)

A. Courses proposed for Diverse Cultures and Critiques/Global Cultures [<Number of proposals, can be NONE>] B. Courses proposed for Diverse Cultures and Critiques/Social Inequality [<Number of proposals, can be NONE] C. Courses proposed for Scientific Investigation [<Number of proposals, can be NONE] D. Courses proposed for Quantitative Reasoning [Number of proposals, can be NONE]

Proposals:

A) HUMANITIES:

<Humanities Division Compiled Department Proposals>

B) NATURAL SCIENCES:

<Natural Science Compiled Department Proposals>

C) SOCIAL SCIENCES:

<Social Sciences Compiled Department Proposals>

D) INTERDISCIPLINARY PROGRAMS

<Interdisciplinary Compiled Department Proposals>

E) GENERAL STUDIES

<Proposal Group #1 Compiled Proposals>

<Proposal Group #2 Compiled Proposals>

<etc...>

F) CRITICAL PERSPECTIVE COURSES

<Critical Perspective Proposals>

1.2.2 Level 3 - Division Compiled Department Proposals

<DIVISION NAME>

1. <Department #1 Proposals>

2. <Department #2 Proposals>

3. <etc...>

1.2.3 Level 2 - Compiled Department Proposals

2. <Department Name>

The <Department Name> proposes <List department proposals>

A. <PROPOSAL #1>

B. <PROPOSAL #2>

C. <etc...>

1.2.4 Level 1 - Proposal

1.3 Proposal Types:

The following is a list of all possible proposals that a faculty member may submit to the Committee on Instruction for approval: (X = complete)

- | | |
|--|---|
| - Add a New Course | ✓ |
| - Revise an Existing Course | ✓ |
| - Drop an Existing Course | ✓ |
| - Add a Critical Perspective to an Existing Course | X |
| - Revise a Major description/requirements | X |
| - Revise a Minor description/requirements | X |
| - Add a New Thematic Minor | X |

Please check the appendix for specific proposal examples.

1.3.1 Add a New Course:

Template:

A. PROPOSAL TO ADD A COURSE TO THE COURSE CATALOG

The <Department> proposes the following new course.

ADD: <Course_Id>: <Course Title>

<Course Description>

Prerequisite: <Course Prereqs>

Rationale: <Rationale>

Library Impact: <Library Impact>

Technology Impact: <Tech Impact>

1.3.2 Revise an Existing Course:

Template:

A. PROPOSAL TO CHANGE <PROPOSED CHANGE CRITERIA>

The <Department> proposes changing the following <Proposed Change Criteria> of **<Current Course Id>**: **<Current Course Title>**

CURRENT <PROPOSED CHANGE CRITERIA>

<Current Course Id>: **<Current Course Title>** <Current Course Description>

Prerequisite: <Current Course Prereqs>

PROPOSED <PROPOSED CHANGE CRITERIA>:

<Proposed Course Id>: <Proposed Course Title> <Proposed Course Description>

Prerequisite: <Proposed Prereqs>

Rationale: <Rationale>

Library impact: <Library Impact>

Tech impact: <Tech Impact>

1.3.3 Drop an Existing Course:

Template:

B. PROPOSAL TO DROP A COURSE

The <Department> proposes dropping the following course.

DROP: <Course Id>: <Course Title> <Course Description>

Rationale: <Rationale>

1.3.4 Major and Minors:

Note: Proposals to revise, add, and remove majors or minors are far less common than proposals to revise, add, or remove a course. Proposals involving majors and minors often involve entire departments, require far more user input, and are more strictly reviewed by the

committee on instruction.

1.3.5 Revise a Major or Minor:

Template:

B. CHANGES TO THE MAJOR/MINOR *(For Faculty/Staff Information)*.

The <Name of Major> Program submits the following changes to the major/minor.

Current Major:

- <Current Major/Minor Requirement 1>
- <Current Major/Minor Requirement 2>
- <etc..>

New Major:

- <Proposed Major/Minor Requirement 1>
- <Proposed Major/Minor Requirement 2>
- <etc..>

Total required courses: <Proposed total required courses>

Rationale: <Rationale>

Library Impact: <Library Impact>

Technology Impact: <Tech Impact>

1.3.6 Add a New Thematic Minor:

The Curriculum Executive Committee proposes adding a <Name of Thematic Minor> with the approval of General Studies and the Committee on Instruction.

1. Name of Minor: <Name of Minor> **2. Name of person submitting the proposal:** <Proposer

Name> **3. Date of proposal:** <Proposal Date> **4. Overview (provide a brief overview of the thematic minor, its purpose and approach. To be published on the CC website as the description of the thematic minor):**

<>

5. What is the purpose of the thematic minor?

<>

6. How are the required courses coherent? How is the thematic minor more than a sampling of courses with related content?

<>

7. How does the thematic minor address specific needs of CC students?

<>

8. How does this thematic minor provide something more than is currently offered in the college's curriculum?

<>

9. Provide a brief history of the conception and development of the thematic minor:

<>

10. Provide at least three assessable (measurable) learning objectives for the thematic minor:

<>

11. Clearly describe the requirements for the

thematic minor:

<>

12. List all currently offered courses that would meet each requirement. For each course, note how often it has been taught per year in the last two years:

<>

13. List any new courses being developed to meet each requirement. For each course, note how often it is planned to be taught over the next two years:

<>

14. List all faculty members and the expertise that qualifies each to teach in the thematic minor:

<>

15. Note any special information, such as those who are permanent v. rotating, etc.:

<>

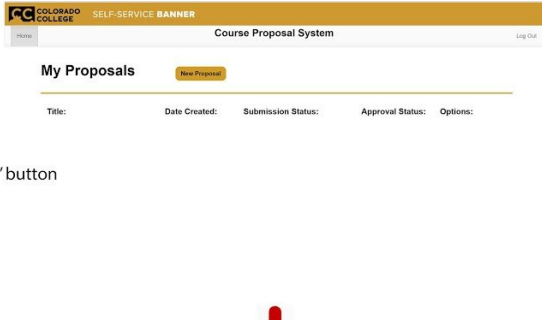
16. This section is optional, but may include vision for the future of the thematic minor, or any other information that explains elements not requested above:

<>

1.4 Create a Web Proposal Walkthrough

1.4.1 Add a New Course Proposal

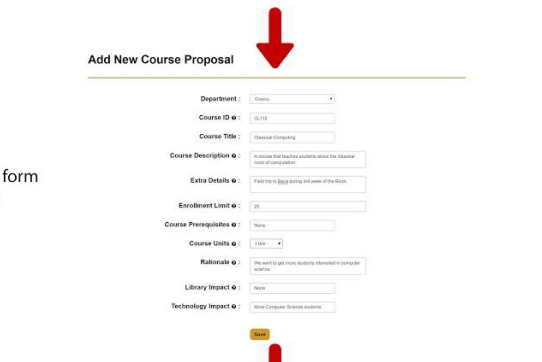
Step 1 - Click "New Proposal" button on Home Page



Step 2 - Select "Add a New Course" from the dropdown menu



Step 3 - Fill out the Proposal form and click "Save" button




Step 4 - Your new Proposal will appear on your Home page

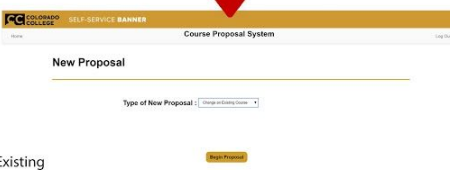


1.4.2 Change an Existing Course Proposal

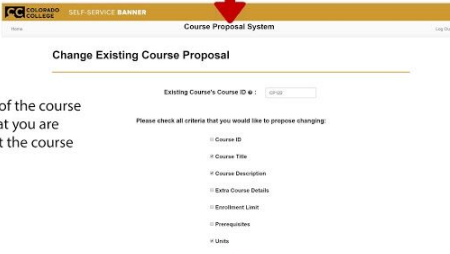
Step 1 - Click "New Proposal" button on Home Page



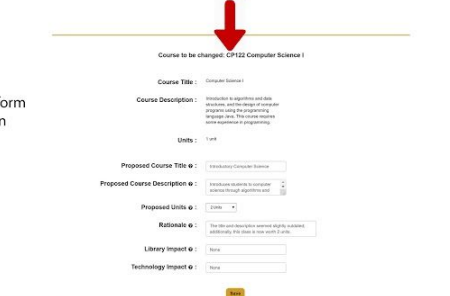
Step 2 - Select "Change an Existing Course" from the dropdown menu




Step 3 - Enter the Course ID of the course to change, as well as what you are proposing to change about the course



Step 4 - Fill out the Proposal form and click the "Save" button



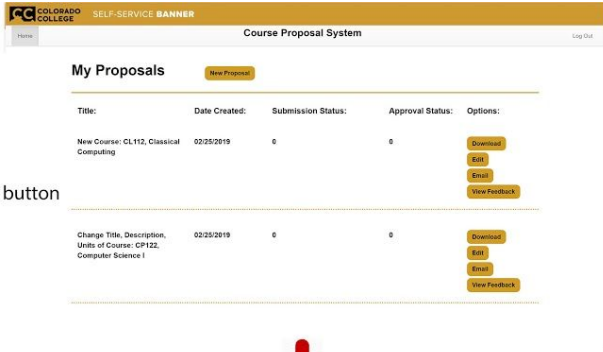
Step 5 - Your new Proposal will appear on your Home page



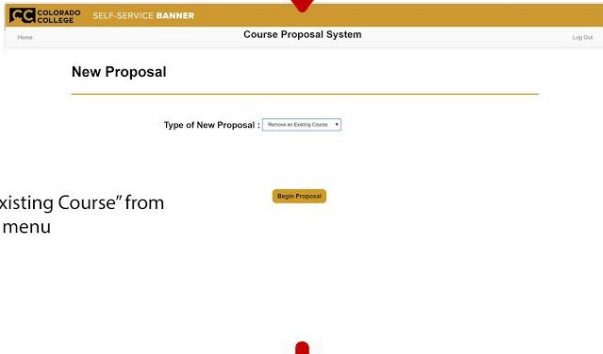
Title:	Date Created:	Submission Status:	Approval Status:	Options:
New Course: CL112, Classical Computing	02/05/2019	0	0	Cancel Save Submit View/Withdraw
Change Title, Description, Units of Course: CP102, Computer Science I	02/05/2019	0	0	Cancel Save Submit View/Withdraw

1.4.3 Remove an Existing Course Proposal

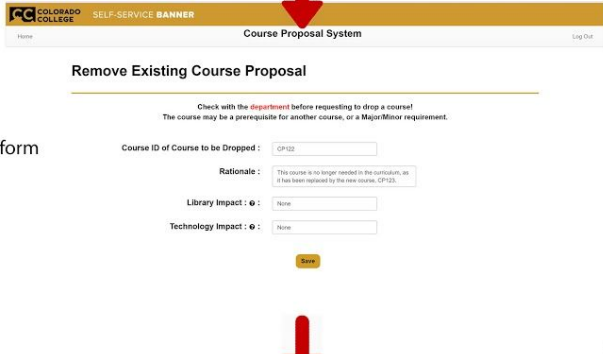
Step 1 - Click "New Proposal" button on Home Page



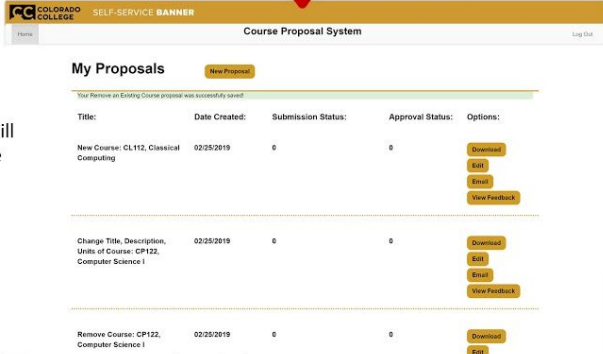
Step 2 - Select "Remove an Existing Course" from the dropdown menu



Step 3 - Fill out the Proposal form and click "Save" button



Step 4 - Your new Proposal will appear on your Home page



Title:	Date Created:	Submission Status:	Approval Status:	Options:
New Course: CL112, Classical Computing	02/25/2019	0	0	Download Edit Email View Feedback
Change Title, Description, Units of Course: CP122, Computer Science I	02/25/2019	0	0	Download Edit Email View Feedback
Remove Course: CP122, Computer Science I	02/25/2019	0	0	Download Edit

1.4.4 Edit a New Course Proposal

Step 1 - Click "Edit" next to "New Course" proposal

Title:	Date Created:	Submission Status:	Approval Status:	Options:
New Course: CL112, Classical Computing	02/25/2019	0	0	Download Edit Email View Feedback
Change Title, Description, Units of Course: CP122, Computer Science I	02/25/2019	0	0	Download Edit Email View Feedback
Remove Course: CP122, Computer Science I	02/25/2019	0	0	Download Edit

Step 2 - Determine what changes you want to make

Edit Proposal "New Course: CL112, Classical Computing"

Current Department:
Current Course ID:
Current Course Title:
Current Course Description:
Current Extra Details:
Current Enrollment Limit:
Current Course Prerequisites:
Current Course Units:
Current Rationale:
Current Library Impact:
Current Technology Impact:

Step 3 - Make the changes and click "Save" button

Edit Proposal "New Course: CL112, Classical Computing"

Current Department:
Current Course ID:
Current Course Title:
Current Course Description:
Current Extra Details:
Current Enrollment Limit:
Current Course Prerequisites:
Current Course Units:
Current Rationale:
Current Library Impact:
Current Technology Impact:

Step 4 - Your edits are saved to the proposal

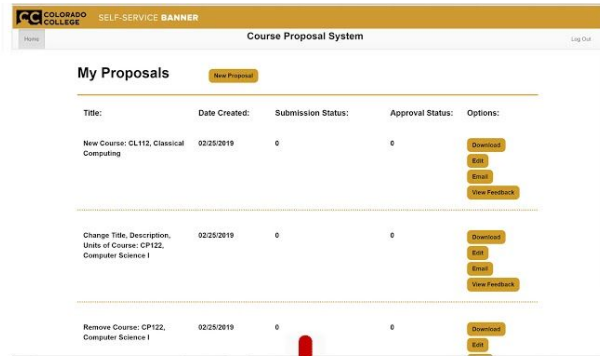
Edit Proposal "New Course: CL112, Classical Computing"

Your edits were successfully saved.

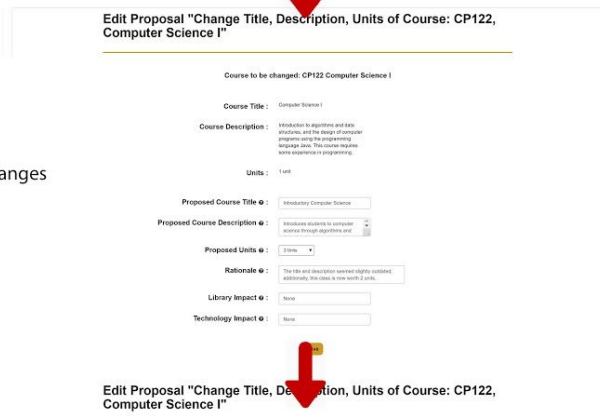
Current Department:
Current Course ID:
Current Course Title:
Current Course Description:
Current Extra Details:
Current Enrollment Limit:
Current Course Prerequisites:
Current Course Units:
Current Rationale:
Current Library Impact:
Current Technology Impact:

1.4.5 Edit a Change Existing Course Proposal

Step 1 - Click "Edit" next to "Change" proposal



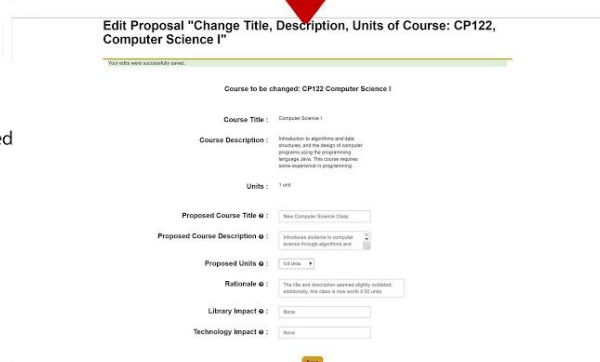
Step 2 - Determine what changes you want to make



Step 3 - Make the changes and click "Save" button

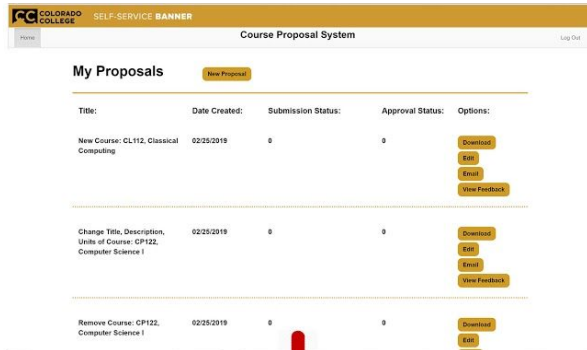


Step 4 - Your edits are saved to the proposal

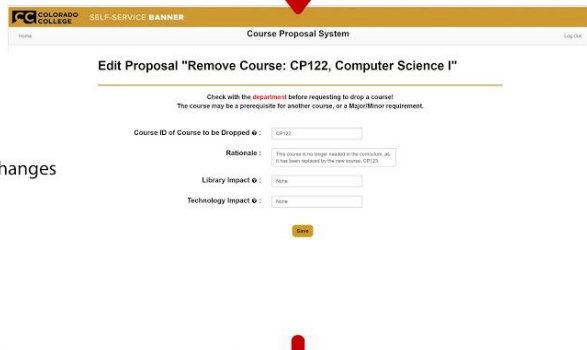


1.4.6 Edit a Remove Existing Course Proposal

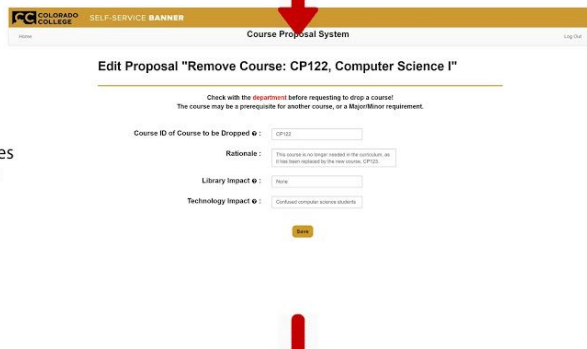
Step 1 - Click "Edit" next to "Remove" proposal



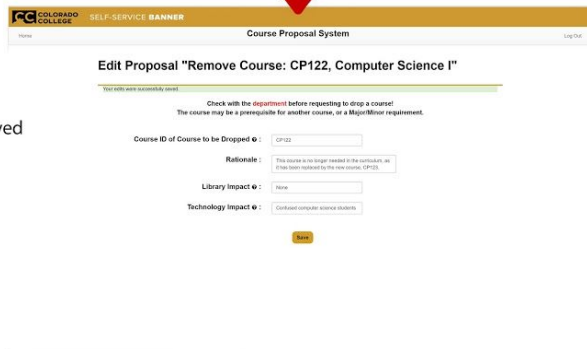
Step 2 - Determine what changes you want to make



Step 3 - Make the changes and click "Save" button



Step 4 - Your edits are saved to the proposal



1.5 Current Feature & Further Implementation

1.5.1 Our Current Feature

This project has been a large undertaking, once all of the potential features for an application like this have been considered. Our system is not only an infrastructure that makes it easier for faculty to create, store, and properly format proposals, but it is also for submitting proposals for approval within the hierarchy of the college. Within the current process for writing and submitting proposals, proposals can take a few different forms relating specifically to courses:

- **Add a New Course** ✓
- **Revise an Existing Course** ✓
- **Drop an Existing Course** ✓

There are also specific changes that can be requested, such as to:

- **Add a Critical Perspective to an Existing Course** X

Proposals can also be written and submitted for reasons relating to Majors and Minors within Departments at the college, such as:

- **Revise a Major Title/Description/Requirements** X
- **Revise a Minor Title/Description/Requirements** X
- **Add a New Thematic Minor** X

During this block, we strove to understand the complicated workflow involved in the current proposal approval process, as well as to determine where we could attempt to

make things simpler. While we have managed to determine the many different proposal types that faculty may need to utilize, we have not been able to implement them all; we will only have time to implement the “**Add a New Course,**” “**Revise an Existing Course,**” and “**Drop an Existing Course**” proposals. Our system will be able to create, store, display, and correctly format each of these proposal types, streamlining the information-gathering and document-formatting process for professors/faculty.

1.5.2 What is Coming Next

The remaining proposal types will ideally be implemented by the next team to take on this project. In the meantime, professors will still be able to use the functionality in our web-app to streamline the creation of most course-related proposals.

Below, we have provided explanations for each of the proposal types that we were not able to implement:

- **Add a Critical Perspective to an Existing Course**
Some courses satisfy certain General Education requirements, such as “Quantitative Reasoning” or “Global Cultures.” In order for a class to satisfy one of these requirements, a faculty member must submit a proposal to add a Critical Perspective to a specific course. These proposals must satisfy certain requirements for each different Critical Perspective. Examples of an “Add a Critical Perspective to an Existing Course” proposal format document can be found in [this document](#).
- **Revise a Major Title/Description/Requirements**
Faculty have the option to propose changes to existing Majors, such as wanting to change a required class or alter the title of the major. An example of this kind of proposal can be found above, in the **Proposal Types** section of this document.
- **Revise a Minor Title/Description/Requirements**
Faculty have the option to propose changes to existing Minors, such as wanting to change a required class or alter the title of the minor. An example of this kind of proposal can be found above, in the **Proposal Types** section of this document.
- **Add a New Thematic Minor**
Faculty have the option to propose a new Thematic Minor for the college to include in its curriculum. These proposals are complicated and contain many

parts; an example of this kind of proposal can be found above, in the **Proposal Types** section of this document.

Beyond the different possible types of proposals that professors/faculty may create for submission, there were a number of other features we wanted to include in our web-app, such as:

- **Option to Edit an Existing Proposal** ✓
- **Download a Proposal as a formatted Word Document** ✓

- **Request Feedback on a Proposal from a colleague** X
- **Give Feedback on a Proposal to a colleague** X
- **Save a Series of Proposals in a single Word Document** X

With our time restraints and the steep PHP/server configuration learning-curve that this project involved, we have so far managed to implement the “**Option to Edit an Existing Proposal**” and the “**Download a Proposal as a formatted Word Document**” features. When editing an existing proposal, the user is able to return to the form they initially filled out, with the information they had submitted in the proper text-boxes, in order to efficiently make changes to the proposal. By clicking the “Download” button next to an existing proposal on the Home page, the user can download a Word Document version of the proposal either for their own records or to send to a colleague to receive feedback.

We hoped that we would have time to implement a Feedback system built directly into the app. This would involve giving users the option to **Request Feedback on a Proposal from a colleague**, who would then see your proposal-feedback request in some sort of feedback-inbox, allowing them to **Give Feedback on the Proposal to the colleague**.

We also hoped to be able to implement the ability to **Save a Series of Proposals in a single Word Document**, where the user could check a box next to multiple proposals and then click “Download” to get a Word Document containing all of the desired proposals.

Unfortunately, these features would take far too long for us to implement at this stage in the process, and the base functionality of our application is not lost without these features. We have chosen to improve the base functionality of our web-application to be 100% tested and functional, rather than attempt to build out more features that we would not have the time to completely validate/test.

2. Developer Guide

2.1 User Stories

Below is the list of User Stories that we identified for this project. They range in users, from Professor to COI divisional representative. The color-coding shows what User Stories we have so far managed to complete, as well as how we ranked them in terms of our priorities:

2.1.1 Completed:

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 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.

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.

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.

Change Course Enrollment Limit

As a professor, I want to be able to change the number of students who can enroll in the course in order to adjust the class setting based on teaching needs or resource limitation.

2.1.2 In Progress:

Check Proposal Submission Deadline

As a professor, I want to be able to check the deadline for submitting my proposals in order to get the proposals approved and brought to the meetings in time.

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.

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.

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.

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.

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.

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.

2.1.3 Icebox:

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

Easily Rebuild

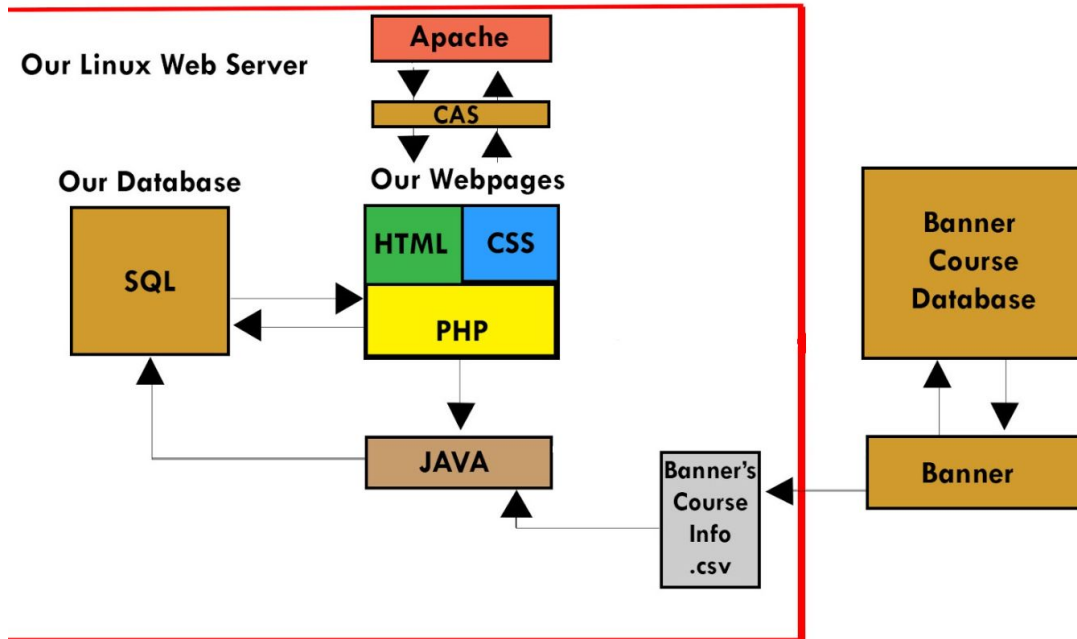
As a software developer, I want to be able to easily rebuild the entire system so that I can fix it when the entire system breaks.

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.

2.2 System Architecture:

This is an overview of our high-level system architecture:



2.2.1 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

2.2.2 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 **proposal-tool 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, the **ExtractExcelData** program reads a .csv file containing the course catalog data, and pushes that information into our **proposal-tool SQL database**. In addition, **Java** is also used to produce a .csv file containing all accepted changed, which is sent to **Banner**.

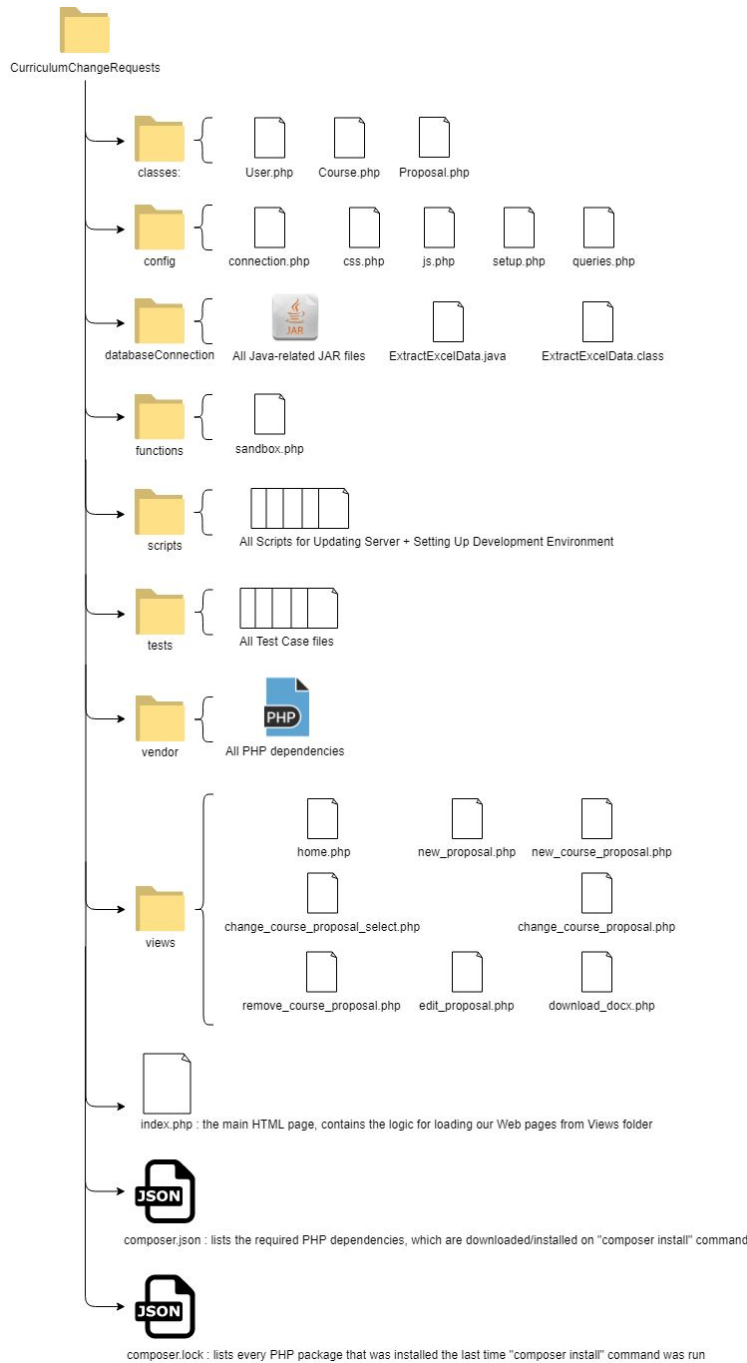
Cron:

Cron is used to schedule the **ExtractExcelData** program to run once per day.

2.3 File System Description

Below is a breakdown of the filesystem we utilized for building this web-application:

Team Software Project - Streamline Course Proposals
 Final Demo Documentation
 Christian Kennedy, Harrison Selle, Jia Kang



Here is an explanation of each of the directories, as well as each file in the directory:

- **Classes** : contains the Class .php files for each of our three classes (User, Course, and Proposal)
- **Config** : contains the files needed to configure the system, such as the database connection (in **connection.php**), the setup of all important variables and importing of

necessary files (in **setup.php**), all of our necessary Javascript code (in **js.php**), all of our CSS styling for HTML elements (in **css.php**), and the back-end functionality that is implemented whenever a form is submitted from a web-page (in **queries.php**).

- **databaseConnection** : this contains all necessary **JAR files** for running the Java-related functions of our application (i.e. updating our database via a .csv file from the Banner database). **ExtractExcelData.java** and **ExtractExcelData.class** are used to read in a .csv file, and then write to the Courses table in our application's MySQL database.
- **Functions** : contains any functions that aren't necessarily class related, such as reading the current web-page's slug (i.e. the identifier for the web-page). These functions are contained inside the **sandbox.php** file; originally this folder also contained our query-related functions, but that implementation was instead moved into the related Class for each function.
- **Scripts** : these files are bash scripts, some of which are run in order to keep our server's code up-to-date with the code in our Git Repository, and the rest of which are used to configure the development environment we utilized during this project.
- **Tests** : contains all of the test files for ensuring that the functionality of our web-application is as expected.
- **Vendor** : contains all of the PHP dependencies necessary for our web-application to function properly, such as PHPWord.
- **Views** : contains the HTML structure for our each of our web-application's web pages.

2.4 Validation

This past week, we worked on a lot of different things to continue getting our app's functionality off the ground. However, beyond implementing user functionality, an important part of delivering any piece of functional software is the ability to **test** that the software does what is expected of it. The most common way to do this is through Unit testing, where you test specific functions to ensure that they return the desired result.

We planned to use PHPUnit for our back-end PHP testing, as PHPUnit is a very well known Unit testing framework for PHP. In preparation for testing, we went back through all of the back-end PHP code we wrote over the past few weeks and re-structured it to utilize Class structures, with specific functions that query our database for specific information whenever necessary. This resulted in much cleaner, more organized code within our filesystem.

While investigating PHPUnit, we came across **Codeception**, a PHP testing framework that utilizes PHPUnit for all back-end testing, and can additionally test front-end capabilities of a web-app like ours.

2.4.1 Back-End Testing

Codeception is built off of the back-end framework of PHPUnit, and utilizes many of the same tests and assertions. The main benefit of using Codeception for our back-end testing rather than PHPUnit is its documentation; PHPUnit's documentation was not only hard to follow, but different versions were spread across the internet, making it difficult to figure exactly how best to tackle setting up PHPUnit tests. Codeception has easy to follow examples of generating Unit tests, and is much easier to actually run when you are ready to test.

Another huge benefit to using Codeception over PHPUnit is the way that Codeception includes not only Unit tests, for ensuring that our basic logic works, but also Integration tests, which don't require that the code being tested be run in isolation. This means that we can test back-end functionality such as querying our MySQL database to add a record to it, then querying it again

to ensure that the record we wanted to test adding was successfully added. This allows for much simpler, more dynamic test creation than PHPUnit's framework, as well as allowing us to test things such as whether or not our MySQL connection is live or not.

By utilizing Codeception's back-end framework, we will be able to ensure that:

1. All of our User-related functions work properly and successfully query our database
2. All of our Course-related functions work properly and successfully query our database
3. All of our Proposal-related functions work properly and successfully query our database

We plan to have all of these tests implemented and passing for 100% of our Class-related testcases by the end of this weekend, in time for our final Software Demo on Tuesday.

2.4.2 Front-End Testing

Codeception's front-end PHP testing framework allows the test running to navigate to web pages, click buttons, and fill out forms, all with the goal of receiving the correct responses and redirects from the web page itself. By utilizing Codeception's front-end framework, we will be able ensure that:

1. All of our web pages load correctly when visited
2. Any invalid URL's redirect to the Home page
3. All of our buttons redirect the user to the correct location when clicked
4. All of our form submissions are processed correctly, without returning errors
5. The user's Home Page correctly displays all of the proposals that they have created/edited

We plan to have all of these tests implemented and passing for 100% of our actual User-related testcases by the end of this weekend, in time for our final Software Demo on Tuesday.

3. Deployment Guide

3.1 Local Development Environment

Our team built a development kit containing all components necessary to run our web app on a local server.

Follow these steps to set up the development kit

3.1.1 Step 1 - Create a virtual machine running Ubuntu 16.04

Note: You may skip this step if you already have a machine with Ubuntu 16.04

- Download and install virtualbox
- Download Ubuntu 16.04
- Create a virtual machine running Ubuntu 16.04 in virtualbox
 - Allocate at least 15 gb of storage

If you want to **work in the virtual machine:**

- Allocate at least 4gb of RAM and 2 cpu cores to your virtual machine

If you want to **work in the host OS:**

- Configure a shared folder to move files from your host OS to the virtual machine

3.1.2 Step 2 - Clone the repository

Inside the terminal of your virtual machine (henceforth devkit) run the following command to clone the repo:

```
$ git clone https://github.com/CP499ColoradoCollege/CurriculumChangeRequests.git
```

3.1.3 Step 3 - Run `./buildDevKit.sh` to install devkit software

Go into the new project directory and type in your terminal:

```
$ chmod +x ./builddevkit.sh
```

```
$ ./builddevkit.sh
```

This bash script will install the following pieces of software automatically:

- **XAMPP**
 - An Apache distribution which includes mySql, php and perl
 - This is the software that will host and serve your **local development server**
- **Java 8**
 - Java is used to process data from banner. See **Architecture Specification**
- **Php extensions, libraries, and Composer**
 - **Php extensions:**
 - php-xml
 - php-zip
 - php-mbstring
 - **Composer libraries:**
 - zend-escaper
 - zend-stdlib
 - PhpWord
 - PHPUnit

Note: Composer is a piece of software that manages php dependencies. For more information:
<https://getcomposer.org/>

3.1.4 Step 4 - Configure XAMPP

XAMPP has a fun feature where it does not show up in your applications. To make it show up there, run the following commands in your devkit terminal:

```
$ sudo apt-get install gksu
```

```
$ gksu gedit /usr/share/applications/xampp-control-panel.desktop
```

gksu will open a text editor. Paste the following text in and save.

[Desktop Entry]

Encoding=UTF-8

Name=XAMPP Control Panel

Comment=Start and Stop XAMPP

Exec=gksudo /opt/lampp/manager-linux-x64.run

Icon=/opt/lampp/htdocs/favicon.ico

Categories=Application

Type=Application

Terminal=false

Now, you can open XAMPP from your applications folder.

3.1.5 Step 5 - Launch XAMPP server

If it is not already open, open the XAMPP control panel by clicking the icon in the top left of your Ubuntu taskbar and typing XAMPP. The XAMPP control panel should be the first result.

In the XAMPP control panel, open the **Manage Servers** tab and click **Start All**

You may now open Firefox in your devkit and type localhost:80 into the navbar to see you a XAMPP welcome page.

3.1.6 Step 6 - Configure mySQL

Navigate to localhost:80/phpmyadmin to open the mysql management gui.

Here, import the database in the repository:

- Create a new database by clicking “New” in the top left corner.
- Select that database in the left hand side, then click “Import” on the toolbar at the top.
- Click “Browse”, navigate to the repository and click proposal-toolDB.sql.
- Click “Go” at the bottom of the webpage to complete database migration.

XAMPP tips:

Project files go in the folder opt/lampp/htdocs from the console, you can click **Go to Application Folder** to open that folder in the file browser. Alternatively, you can get there from the command line with `cd ../../../../opt/lampp/htdocs`

3.1.7 Step 7 - Run ./deploy_development.sh

In the repository, there is a script called `./deploy_development.sh`. Running the script will perform the following:

- Copy **php**, **html**, **css** files to the local server web root located at `~/../opt/lampp/htdocs`
- Run `composer install` to update php libraries in `~/../opt/lampp/htdocs`

Done!

You are ready to begin development on the local server!

You may either work directly in the `opt/lampp/htdocs` directory to have your changes instantly update the web server at `localhost:80`, or, you may work in another directory and run the `./deploy_development.sh` script to push changes to the local web server (Recommended)

3.2 Deployment

Our web server is hosted at <https://proposal-tool.coloradocollege.edu>

The server is managed by ITS. **If you have any technical problems deploying to the server contact ITS early and often**

3.2.1 Step 1 - Get access to the server

Contact ITS about getting access to the server.

3.2.2 Step 2 - Run `./deploy_server.sh`

This script does the following:

- Compiles **ExtractCourseInfo.Java**
- Transfers **php**, **html**, **css**, and **compiled java** to the web server using **sftp**
- **SSH** into the server to :
 - Move **php**, **html**, and **css** files to the web root located at `~/../var/www/html`
 - Runs **composer install** to update **php** libraries on server

Done!

Your code is now deployed. Be sure to visit <https://proposal-tool.coloradocollege.edu> to verify everything is working.

Note: You must be on the **CC network** and use **https**.

3.3 Maintenance

3.3.1 Data stream

Our web application is ready to integrate with the Banner DB via a .csv stream. This stream can be handled by our java program called **ExtractCourseInfo.java**, located in the databaseConnection directory. The **.deploy_server.sh** script should start the **cron** service, which runs **ExtractCourseInfo** once per day.

ExtractCourseInfo relies on a .csv containing up-to-date course catalog information. Currently, we have no system in place to obtain that up-to-date file on a daily basis. We were given one of these such documents at the beginning of our project. It is called example.xlsx and it is located in the databaseConnection directory.

3.3.2 Software updates

The **deploy_server.sh** script will update php libraries by running `composer install`. This command installs each library listed in the **composer.json** file, as well as their dependencies.

Important: If you need additional php libraries, you must run `composer require <package name>` **on the server via ssh**. This command will add that package to **composer.json**, so that they may be installed with `composer install`.

Additional composer help: <https://www.engineyard.com/blog/composer-its-all-about-the-lock-file>

Always keep packages consistent between your development environment and the server. An easy way to ensure this is to run the following command on both the development environment and the server:

```
sudo apt update
```

```
sudo apt upgrade
```

This should update each non-composer package used in this project to their most up-to-date stable releases.

APPENDIX A

Proposal Types Examples

A.1 Add a New Course

A. PROPOSAL TO ADD A COURSE TO THE COURSE CATALOG

The Department of Chemistry and Biochemistry proposes the following new course.

ADD: CH471: Ribonucleic Acids.

This class covers the structure and function of RNA from a biochemical perspective. There are many different large and small RNA that are present in the cell that perform key functions in the cell from splicing, protein synthesis, to regulation. Structure and function of RNA and the techniques used to study these will be discussed using current literature. Biological functions of ribozyme and non-coding RNA will be studied with an eye towards understanding the development of new techniques in molecular biology for artificial manipulations of cellular systems, drug development, and human genome manipulation. Ethical challenges associated with RNA-based technologies will also be discussed. The course is based in current literature with substantial independent and group learning components. A research-based laboratory is included.

Prerequisite: CH382 or COI. 1 unit.

Rationale: As of the 2017-2018 Academic Year, a version of this course has been offered 3 times as a special topics course (CH 400, Advanced Topics in Chemistry: Nucleic Acid Chemistry or Ribonucleic Acid Chemistry). Continued offering requires an official course designation. The course has been well received by students and the Chemistry and Biochemistry Department would like to make it a permanent addition to our offerings. CH 471 – Ribonucleic Acids is intended to provide a deeper investigation into the macromolecular structure and function. The course expands upon the protein structure and function concepts that were introduced in CH 382 – Biochemistry I. The curricular changes that were introduced

by the Chemistry and Biochemistry department in the academic year 2010-2011 allows students increased flexibility to take advanced topical courses in their particular field of chemistry that count towards the major as electives. This course will provide Chemistry and Biochemistry majors an additional opportunity to delve deeper into the chemistry of biological processes. Biological science majors have taken these courses as it is on a current topic not covered extensively by other departments.

Library Impact: No impact. The course has been taught three times as a topics course and has not needed any new library resources. The resources needed were already available because they overlap with what we use for Biochemistry I and II.

Technology Impact: None.

A.2 Revise an Existing Course

A. PROPOSAL TO CHANGE COURSE DESCRIPTION AND PREREQUISITES

The Department of Mathematics and Computer Science proposes changing the following course description and prerequisites of **MA125: Precalculus and Calculus**.

CURRENT COURSE DESCRIPTION AND PREREQUISITE

MA125: Precalculus and Calculus This course covers the same material as 126 together with one block of content from algebra, trigonometry, analytic geometry and the study of functions. Intended solely for students not sufficiently prepared for 126. (Fulfills one unit of the divisional requirement in the natural sciences. Meets the Critical Perspectives: Scientific Investigation of the Natural World requirement. Meets the Critical Perspectives: Quantitative Reasoning requirement.)

Prerequisite: Consent of Instructor.

PROPOSED COURSE DESCRIPTION AND PREREQUISITE:

MA125: Precalculus and Calculus This course covers the same material as MA126 together with one block of content from algebra, trigonometry, analytic geometry and the study of functions. Intended solely for students not sufficiently prepared for MA126. (Meets the Critical Perspectives: Scientific Investigation of the Natural World requirement. Meets the Critical

Perspectives: Quantitative Reasoning requirement.)

Prerequisite: none.

Rationale: The divisional requirement no longer exists; therefore, we propose to remove “Fulfills one unit of the divisional requirement in the natural sciences” from the course description. The course required “consent of instructor” in the past to help steer students who were not prepared to take MA126 (Calculus 1) toward MA125. In recent years, and with the help of the QRC Director, Steve Getty, and the Advising Hub, it appears that placement through advising is sufficient, so that obtaining the consent of instructor becomes an unnecessary hurdle in enrolling in the class. Therefore, we propose to remove the “consent of instructor” prerequisite.

Library impact: none

A.3 Drop an Existing Course

B. PROPOSAL TO DROP A COURSE

The Department of Education proposes dropping the following course.

DROP: ED201: Advanced Aids in Colorado Springs Schools

Serve as an advanced instructional aide in local schools, completing 30 hours of service-learning in cooperation with local education personnel. Activity varies according to the needs of the placement, but the emphasis is on gaining a deeper understanding of lesson planning and teaching a lesson with the focus on differentiation under the guidance of the placement personnel.

Rationale: This course is no longer needed, as students are obtaining necessary experiences in existing courses.

A.4 Revise a Major/Minor

B. CHANGES TO THE MAJOR (*For Faculty/Staff Information*).

The Environmental Studies Program submits the following changes to the major.

Current Major:

- EV145: Environment and Society
- EV128: Intro to Global Climate Change
- MA125: or 126: Calculus
- One 100 or 200 level Environmental Science Course
- EV271: Environmental Law & Policy OR EV274: Environmental Politics & Policy
- One Environmental Justice/Environmental Equity unit: EV274: Cities, Sustainability, and Environmental Justice; SW272: Nature, Region, and Society of the Southwest; EV276: Environmental Sociology; EV277: Ecofeminism; SW220: Environmental Justice in the Southwest
- EC201: Economic Theory 1
- Two 200 level environmental humanities courses: EV281: Environmental Ethics; EV255: Nature and Society; EV285: Introduction to Literature and Environment; HY212: American Environmental History
- Four EV Social Science, Humanities, or Natural Science electives, at least two of which are 300 level. Up to two study abroad courses may count as 200 level electives. Independent study may count on case-by-case basis based on a petition. (Optional -- EV391: Junior Research Seminar - required as one of the four electives for students writing a thesis)
- EV373: Public Policymaking
- EV421: Environmental Synthesis
- EV499: Thesis (optional for the major, but required for distinction)

New Major:

- MA126 or 126: Calculus
- EV128: Introduction to Global Climate Change
- EV145: Environment and Society
- One additional 100- or 200-level Environmental Science, Geology, or Organismal Biology and Ecology course, to be agreed upon with your advisor
- EC201: Economic Theory 1

- One of the following Environmental Policy courses: (1) EV271: Environmental Law and Policy (2) EV274: Environmental Politics and Policy
- One of the following 300-level Policy/Economics courses: (1) EV373: Public Policymaking (2) EV341/EC271: Ecological Economics (3) EV321: Environmental Management
- One of the following Environmental Justice/Environmental Equity courses: (1) EV274: Cities, Sustainability, and Environmental Justice (2) EV276: Environmental Sociology (3) EV277: Ecofeminism (4) SW220: Environmental Justice in the Southwest (WI) (5) EV375: Community Forestry (6) EV/SW301: Political Ecology of the Southwest (WI)
- Two of the following Environmental Humanities courses: (1) EV281: Environmental Ethics (2) EV255: Nature and Society (3) EV285: Introduction to Literature and Environment (4) HY212: American Environmental History
- Four EV Social Science, Humanities, or Natural Science electives, at least two of which are 300- level. Independent study may count on a case-by-case basis by petition. EV391: Junior Research Seminar is required as one of the four electives for students writing a thesis.
- EV421: Environmental Synthesis
- EV499: Thesis (optional for the major, but required for distinction)

Total required courses: 15 (EV499: Thesis is optional)

Rationale: These changes provide students with more options and freedom to pursue topics more closely aligned with the interests and build upon their previous coursework in economics and policy. In addition, these changes enable students to see the importance of curricular symmetry since 200-level options exist for policy, economics, and politics. Furthermore, these changes offer students more options under the environmental justice/environmental equity component of the major, and offers students the opportunity to satisfy the Writing Intensive requirement through Environmental Studies. We are adding options for students hoping to meet the requirements for the major and are creating program flexibility for staffing and sabbatical absences.

Library Impact: None.

Technology Impact: None.